Market Integration of Virtual Power Plants
PhD Thesis

November 2014

ISBN: 978-87-7152-057-6

Contents

Preface and Acknowledgements

This thesis is submitted as a collection of papers in partial fulfilment of the requirements for the degree of Doctor of Philosophy at the Section of Automation and Control, Department of Electronic Systems, Aalborg University, Denmark. The work presented in this thesis is carried out under the Industrial PhD programme supported by the Danish Ministry of Higher Education and Science. The work has been carried out from January 2011 to October 2014 in collaboration with DONG Energy and the Section of Automation and Control, Aalborg University, under the supervision of Associate Professor Jan Dimon Bendtsen, Professor Jakob Stoustrup, Senior Engineer Lars Henrik Hansen, Senior Energy System Architect Kristian Edlund and Owner, Added Values, Tommy Mølbak.

I would like to thank my supervisors for their invaluable support, guidance and inspiration throughout the process of completing the PhD project.

Abstract

Global efforts to reduce emissions of carbon dioxide drives the introduction of renewable power production technologies into the existing power system. The real-time balance between production and consumption must, however, still be maintained at all times. Unfortunately, this is becoming increasingly challenging due to the intrinsic variability of production technologies such as photovoltaics and wind turbines. In a Smart Grid system the balancing task will therefore be handled by mobilizing flexibility on the consumption side.

This Thesis assumes that the Smart Grid should be commercially based rather than funded by subsidies. Consequently the Smart Grid provides a business opportunity for so-called Virtual Power Plants. A Virtual Power Plant is an independent commercial operator, which provides Smart Grid capabilities to flexible consumers. This means that the Virtual Power Plant is the technical and commercial entity in charge of mobilization and control of flexible consumers. This Thesis addresses some of the challenges relating to the Smart Grid and Virtual Power Plant visions with special attention to flexibility, value creation and portfolio coordination.

The term flexibility is central to the Smart Grid discussion, but it is difficult to give a precise definition of flexibility. This Thesis therefore suggests the use of a simple taxonomy for modelling consumer flexibility. The taxonomy consists of three archetypal flexibility models, but it does not exhaust the flexibility term. It does however significantly sharpen the discussion of the flexibility concept and provides a categorization of flexible systems.

This Thesis also investigates what value can be created from the different types of flexibility by assuming that the Virtual Power Plant will generate profit by trading flexibility in electricity markets. Based on the taxonomy and a model of the Nordic electricity markets it is explored how differences between flexibility types affects profit margin. It is found that revenue potential depends strongly on the quality of flexibility.

Finally the subject of portfolio coordination is addressed, since a major challenge in developing the Smart Grid is that thousands or even millions of flexible consumers must be coordinated to operate in a sensible, interconnected manner. Due to the sheer size of the coordination problem, however, the computation time associated with coordination can be problematic. This issue is first investigated through analytical contributions on the circumstances under which portfolio coordination becomes computationally challenging. Next, several options for finding optimal solutions of the coordination problem are investigated, namely use of the software package CPLEX, Dynamic Programming and Dantzig-Wolfe Decomposition. Since non of these efforts scale to large problem instances the option of heuristic optimization is explored. Several methods are investigated and promising results are found both regarding computation time and solution quality.

Synopsis

Den globale indsats for at reducere udledningen af CO_2 medfører, at vedvarende produktionsteknologier bliver introduceret i det eksisterende elsystem. Dermed bliver det stadigt mere udfordrende at opretholde balancen mellem produktion og forbrug på grund af den naturligt svingende produktion fra teknologier som solceller og vindmøller. I et Smart Grid system skal denne balanceringsudfordring løftes ved at mobilisere fleksibilitet på forbrugssiden.

Denne afhandling antager, at Smart Grid systemet skal være kommercielt baseret og ikke blot finansieret af offentlige tilskud. Dermed skaber Smart Grid systemet en forretningsmulighed for de såkaldte Virtuelle Kraftværker. Et Virtuelt Kraftværk er en kommerciel operatør, som leverer Smart Grid funktionalitet til fleksible forbrugere. Dette betyder, at det Virtuelle Kraftværk er den tekniske og kommercielle enhed, der håndterer mobilisering og kontrol af fleksible forbrugere. Denne afhandling adresserer nogle af udfordringerne omkring Smart Grid systemet og de Virtuelle Kraftværker. Fokusområderne er fleksibilitet, værdiskabelse og porteføljekoordinering.

Begrebet fleksibilitet er centralt i Smart Grid diskusionen, men det er vanskeligt at give en præcis definition af fleksibilitet. Denne afhandling foreslår derfor brugen af en simpel taksonomi til modellering af forbrugerfleksibilitet. Den foreslåede taksonomi etablerer tre fleksible arketyper, men er ikke udtømmende for fleksibilitetsbegrebet. Dog nuancerer den diskussionen af begrebet fleksibilitet og fremsætter en mulig kategorisering af fleksible systemer.

Efterfølgende undersøges det, hvilket afkast et Virtuelt Kraftværk kan forvente af hver af de tre fleksibilitetstyper. Dette gøres ved at antage, at det Virtuelle Kraftværk skaber profit ved at handle fleksibilitet på elmarkederne. Baseret på taksonomien og en model af de nordiske elmarkeder, undersøges det, hvordan forskellene mellem fleksibilitetstyper påvirker indtægten. Det findes, at det potentielle afkast er stærk afhængig af kvaliteten af fleksibilitet.

Til sidst adresseres porteføljekoordinering, da en stor udfordring i at udvikle Smart Grid systemet er, at tusinder eller måske millioner af fleksible forbrugere skal koordineres og operere på en fornuftig, sammenkoblet måde. Den blotte størrelse på koordineringsproblemet gør derfor, at beregningstiden forbundet med koordinering kan blive problematisk. Denne problematik undersøges først ved analytiske bidrag omkring hvornår porteføljekoordinering bliver beregningsmæssigt problematisk. Dernæst undersøges forskellige muligheder for at beregne optimale løsninger til koordineringsproblemet. Specifikt undersøges brug af softwarepakken CPLEX, dynamisk programmering og Dantzig-Wolfe dekomposition. Desværre skalerer ingen af disse metoder til store problemer og derfor undersøges heuristisk optimering. Dette giver særdeles lovende resultater både med hensyn til beregningstid og løsningskvalitet.

Thesis Details

Thesis Title:	Market Integration of Virtual Power Plants
PhD Student:	Mette Kirschmeyer Petersen
Company Supervisors:	Senior Engineer Lars Henrik Hansen, DONG Energy
	Senior Energy System Architect Kristian Edlund, DONG Energy
	Owner Tommy Mølbak, Added Values
University Supervisors:	Assoc. Prof. Jan Dimon Bendtsen, Aalborg University
	Prof. Jakob Stoustrup, Aalborg University

The main body of this thesis consists of the following papers.

[1] Mette Kirschmeyer Petersen, Lars Henrik Hansen and Tommy Mølbak , "Exploring the Value of Flexibility: A Smart Grid Discussion", *IFAC Symposium on Power Plants and Power System Control*, vol. 8, no. 1, pp. 43–48, 2012.

[2] Mette Kirschmeyer Petersen, Jan Dimon Bendtsen and Jakob Stoustrup, "Optimal Dispatch Strategy for the Agile Virtual Power Plant", *American Control Conference*, pp. 288–294, 2012.

[3] Mette Kirschmeyer Petersen, Kristian Edlund, Lars Henrik Hansen, Jan Dimon Bendtsen and Jakob Stoustrup, "A Taxonomy for Modeling Flexibility and a Computationally Efficient Algorithm for Dispatch in Smart Grids", *American Control Conference*, pp. 1150–1156, 2013.

[4] Mette Kirschmeyer Petersen, Lars Henrik Hansen, Jan Dimon Bendtsen, Kristian Edlund and Jakob Stoustrup, "Market Integration of Virtual Power Plants", *IEEE Conference on Decision & Control*, pp. 2319–2325, 2013.

[5] Mette Kirschmeyer Petersen, Lars Henrik Hansen, Jan Dimon Bendtsen, Kristian Edlund and Jakob Stoustrup, "Heuristic Optimization for the Discrete Virtual Power Plant Dispatch Problem", *IEEE Transactions on Smart Grid*, Vol. 5, No. 6, pp. 2910–2918, 2014.

In addition to the main papers, the following publications have also been made.

[6] Klaus Trangbaek, Mette Kirschmeyer Petersen, Jan Dimon Bendtsen and Jakob Stoustrup, "Exact Power Constraints in Smart Grid Control", *IEEE Conference on Decision & Control*, pp. 6907–6912, 2011.

[7] Klaus Trangbaek, Mette Kirschmeyer Petersen, Jan Dimon Bendtsen and Jakob Stoustrup,, "Predictive Smart Grid Control with Exact Aggregated Power Constraints", *Smart Power Grids 2011*, Springer Berlin Heidelberg, pp. 655–674, 2012.

This thesis has been submitted for assessment in partial fulfillment of the PhD degree. The thesis is based on the submitted or published scientific papers which are listed above. Parts of the papers are used directly or indirectly in the extended summary of the thesis. As part of the assessment, co-author statements have been made available to the assessment committee and are also available at the Faculty. The thesis is not in its present form acceptable for open publication but only in limited and closed circulation as copyright may not be ensured.

1 | Introduction

1.1 Motivation

Global efforts to reduce CO_2 emissions has driven the introduction of renewable power generation technologies into the power system. However, since these new generation technologies harvest energy from sun and wind the power availability is becoming increasingly changeable and difficult to predict. The Smart Grid is born out of the need to maintain the balance between production and consumption in this far more volatile power system (see Figure 1.1). In the Smart Grid a communication link to the consumption side is to be established, such that flexible consumers like electric vehicles, heat pumps and HVAC-systems can be mobilized, organized and activated to follow power availability and scarcity.

This thesis takes a top-down and commercial approach to the Smart Grid, by assuming that the Smart Grid must eventually become self-financing and that Smart Grid players must develop their business within the existing frame works of deregulated electricity

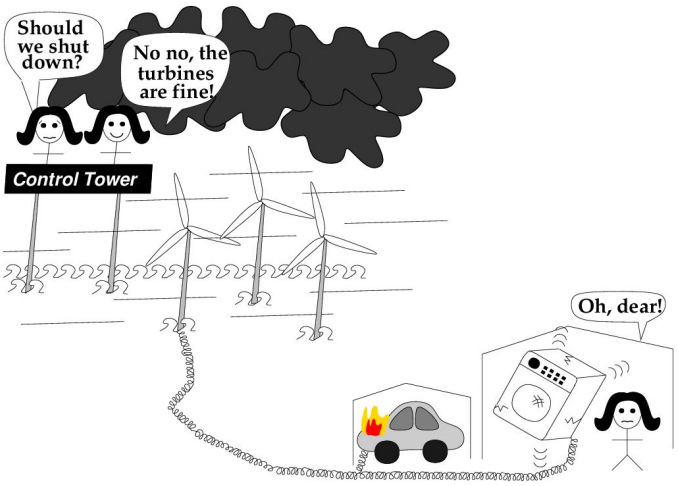

Figure 1.1: Balancing: When electricity is produced it is consumed instantly, so production and consumption must be precisely balanced at all times.

markets [8]. Under these assumptions, however, becoming a part of the Smart Grid is by no means a trivial task.

Consider an asset owner, who owns a system, which is flexible in some way. The asset owner then has an opportunity to become a Smart Grid player by offering flexibility to the power system. The majority of assets have, however, been installed for some primary purpose and a Smart Grid service add on would in most cases be a secondary goal. The motivation for introducing a Smart Grid service add on should be a reduction of electricity costs.

To become a Smart Grid player the asset owner must develop extensive in-house Smart Grid capabilities, such as system modelling and control, establishment of a secure and stable communication structure, development of market forecasting and trading capabilities etc. It can therefore be argued, that *"while transaction costs [...] provides an impetus to move activities inside the boundaries of the firm, [..] envy and resulting social comparison costs motivate moving activities outside of the firm*, [9].

The Smart Grid thus provides a business opportunity for third party aggregators denoted Virtual Power Plants (see Figure 1.2). A Virtual Power Plant is an independent commercial operator, which provides Smart Grid capabilities to asset owners, [10]. This means that the Virtual Power Plant does not have ownership of the flexible systems in its portfolio, but only provides services such as control, communication and/or trading facilities to asset owners. To make a profit the Virtual Power Plant must therefore build a lucrative portfolio of assets and manage this in the electricity markets. Some contract between the Virtual Power Plant and asset owners should then specify how profit is shared, see [11]. With this vision, flexibility becomes a commodity in itself and multiple Virtual Power Plants can compete for flexible assets in order to obtain the most profitable portfolio.

This thesis addresses some of the challenges faced by a Virtual Power Plant with special attention to flexibility modelling, value creation and computational complexity of portfolio coordination.

Flexibility

The flexibility of the aggregated assets is *the* value-adding resource of the Virtual Power Plant, since its total profit gain comes from trading the portfolio in the electricity markets. A first challenge for the Virtual Power Plant is therefore how to model the assets and flexibility in the portfolio.

The flexibility of a given system is intrinsic and both state- and time dependent. It is therefore very difficult to give a formal definition of that all-important Smart Grid keyword: Flexibility. And it is perhaps even more difficult to determine whether one system is more flexible than another.

In this Thesis we therefore propose the use of abstracted flexibility models rather than detailed system models. The concept is similar to how markets operate today: When power producers submit bids to electricity markets they specify commodity relevant data such as time of delivery, activation time, quantity, price and location, [12] and [13]. They *do not*, however, specify system data such as fuel type, generator efficiency or correlation to district heating, [14]. Such details are abstracted from in the bidding process, and this abstraction enable the markets to operate efficiently.

In this thesis flexibility will therefore be described by a simple taxonomy (see Section

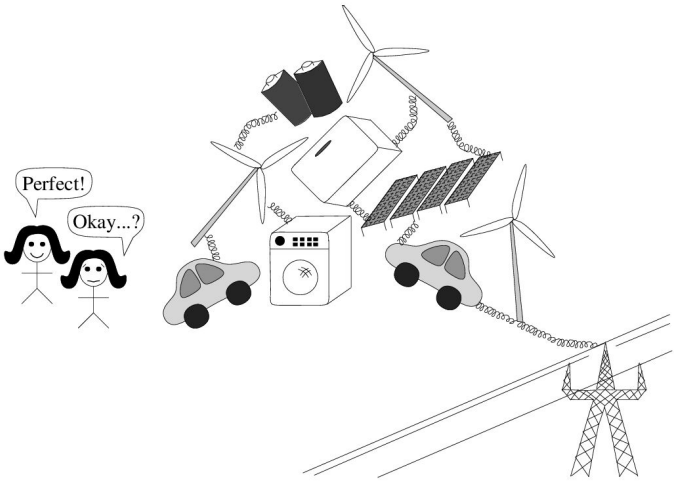

Figure 1.2: Virtual Power Plant: A Virtual Power Plant can pool a large number of consumption units such as cooling systems, heat pumps and electric vehicles in order to generate the flexibility on the consumption side, which is decreasing on the production side.

3.1), rather than detailed system models, such as those given in [15], [16] and [17]. The taxonomy is denoted *Buckets*, *Batteries and Bakeries* and the formulation is a proper taxonomy in the sense that there is a hierarchical relationship between the different types of flexibility. This still does not allow us to state formally whether one system is more flexible than another, but it does allow us to state in a meaningful way that certain types of flexibility is of better quality than others.

Market Integration

To make competition fair Virtual Power Plants must compete on equal terms with other players such as wind farm operators and traditional power plants by offering flexibility through the electricity markets. The market integration of a Virtual Power Plant can be direct, through integration with a higher level Virtual Power Plant or through integration with a larger portfolio of production units, see Figure 1.3. Virtual Power Plants will then help the system wide goal of load balancing simply by increasing the capacity in the markets. Market mechanisms should then generate a utilization of the total available capacity, which is cheaper and more efficient than the current configuration.

In a deregulated electricity market the balance between supply and demand is maintained though the electricity markets all of which are potentially open to Virtual Power Plants. Electricity markets operates in a series one after the other as the time of delivery approaches (see Figure 1.4). On each market producers and wholesalers make bids for future time slots based on the best available knowledge, such as wind forecasts, consumption forecasts or general market knowledge and experience. Electricity prices are then determined based on these bids, and a balanced plan for production and consump-

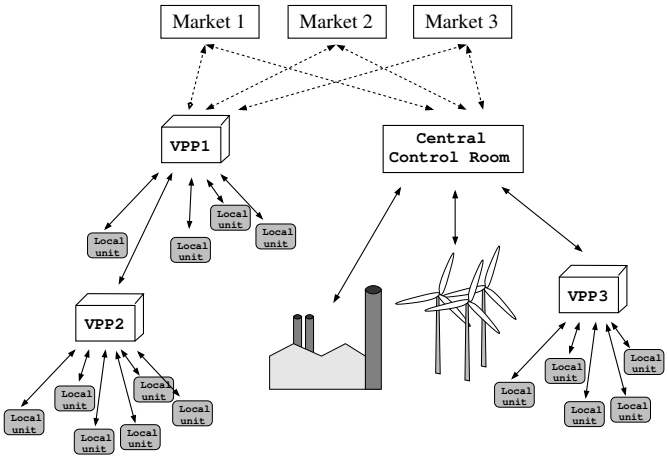

Figure 1.3: The market integration of a Virtual Power Plant can be direct, through integration with a higher level Virtual Power Plant or through integration with a larger portfolio of production units.

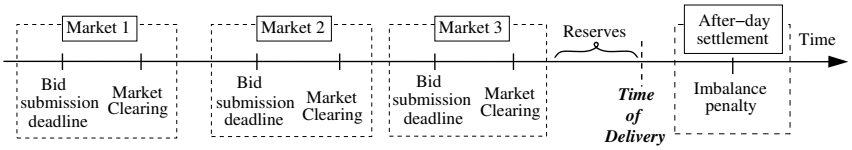

Figure 1.4: Electricity markets operate in several cycles. As the time of delivery approaches reserves take over the balancing task as markets cannot operate fast enough to balance production in real time.

tion is generated. Once prices on an electricity market are settled (Market Clearing) that market is closed.

At some point, however, it is so close to the time of delivery, that ”the price mechanism cannot work fast enough to balance consumption and production in real time” [18]. At this point the so called reserves (ancillary services) take over the responsibility of keeping production and consumption balanced in real time. Traditionally, reserves are provided by power plants, which are operating at less than full capacity. Consequently these power plants can ramp up or down as needed. The task of reserve ”standby” is traded in designated reserve markets. These markets are also open to Virtual Power Plants and could be especially attractive as reserves can be compensated with an availability payment as well as an activation payment (see Section 2.1 for further details).

Portfolio Coordination

A major challenge in developing the Smart Grid is the sheer size of the optimization problems involved. Solving a dispatch problem for a traditional power system with tens or hundreds of generators is a challenge, which has been researched for decades, see [19],

[20], [21]. Moving to the Smart Grid, however, will expand that problem with additional thousands or even millions of units. This means that the computation time associated with determining an optimal dispatch configuration is very likely to be unacceptable in practice, especially because the system must operate in real time.

A Virtual Power Plant does not require the construction of a facility as such, but there are a number of fixed costs, which must be covered in order for the Virtual Power Plant to be profitable. Examples of such expenses are

- marketing, billing and accounting,
- installation and maintenance of metering equipment,
- installation and maintenance of communication equipment and
- development of IT-platform.

To benefit from economies of scale to recover these fixed costs the Virtual Power Plant must therefore accumulate a portfolio of a considerable size. Consequently this thesis will address the challenge of computational complexity in the Smart Grid by investigating portfolio coordination of a Virtual Power Plant.

1.2 Research Questions

The focus of the PhD project has been flexibility, value creation and computational complexity of portfolio coordination for a Virtual Power Plant. These themes are investigated from different perspectives in papers [1] to [5]. In this thesis the main contributions of the project will be presented by investigating the following research questions:

1. **Is a better quality of flexibility also more valuable?**
   Once the taxonomy for defining flexibility has been introduced the concept of quality of flexibility follows directly. It then remains to be verified that a better quality of flexibility is also more valuable. Here more valuable means that it can generate larger revenues when traded in electricity markets.

2. **How can the Virtual Power Plant preserve the quality of the flexibility in the portfolio as flexible units are dispatched?**
   If certain forms of flexibility are both of better quality and higher value than others then the Virtual Power Plant should attempt to maximize the quality of the portfolio during operation. This means that when a portfolio of diverse units are available the worst quality units should be dispatched first. It will be investigated how to achieve this.

3. **Is coordination of a large portfolio of flexible units computationally challenging for the Virtual Power Plant? How can this issue be mitigated?**
   As discussed above the Virtual Power Plant must accumulate a portfolio of a certain size to cover fixed costs. This might however mean that portfolio coordination could become computationally challenging. It will be investigated when this issue becomes critical and what method are available to handle the problem.

Responses to these Research Questions 1 to 3 based on findings from papers [1] to [5] are presented in Chapter 3.

1.3 Outline of the Thesis

This thesis is presented as a collection of papers and it is divided into two main parts: Thesis Details and Contributions.

Thesis Details

The first part of the Thesis, Thesis Details, has already begun with motivation and research questions. Thesis Details is a coherent summary of the main scientific contributions of the PhD project based on Papers [1] to [5].

An additional contribution is presented in Section 3.7 under *Prediction & Agility*. This contribution consists of the novel algorithms *Predictive-Balancing-with-Agility* and *Agile-Balancing-with-Prediction*, which have not been previously published. These algorithms combine prediction and agility, which leads to solutions of far better quality than the algorithms *Agile-Balancing* and *Predictive-Balancing*, which have been published in [3].

Contributions

The second part of the thesis, Contributions, is a presentation of the five publications made during the project. Papers are presented in the following order

- **Exploring the Value of Flexibility: A Smart Grid Discussion, [1]**
  This paper describes the motivation behind the more theoretical contribution of [2]. It thus discusses how the value of flexibility in an electricity system is determined by markets, forecasts and physics. It also reviews the many constraints that are relevant to determining the value of a flexible resource and introduces the ideas of *quality of flexibility* and *agility*.

- **Optimal Dispatch Strategy for the Agile Virtual Power Plant, [2]**
  In this paper it is proved formally that when local units are power and energy constrained integrators with $\underline{P} = \underline{E} = 0$ there exists a dispatch strategy, which is optimal regardless of future load/imbalances. It is also proved that the optimal dispatch can be obtained by solving a quadratic program at each sample.

- **A Taxonomy for Modelling Flexibility and a Computationally Efficient Algorithm for Dispatch in Smart Grids, [3]**
  This paper presents the *Buckets*, *Batteries and Bakeries* taxonomy for modelling flexibility in Smart Grids as well as two heuristic algorithms for solving the balancing task: *Predictive-Balancing* and *Agile-Balancing*. *Predictive-Balancing* is a traditional moving horizon algorithm, where power is dispatched based on perfect predictions of the power supply. *Agile-Balancing*, on the other hand, is strictly non-predictive, but designed to exploit the heterogeneity of the flexible units.

  It is demonstrated that in spite of being non-predictive, *Agile-Balancing* can in some cases out-perform *Predictive-Balancing* even when *Predictive-Balancing* has perfect prediction over a relatively long horizon. This is due to the flexibility-synergy-effects, which *Agile-Balancing* generates. As a further advantage *Agile-*

Balancing is extremely computationally efficient since it is based on sorting rather than solving a linear program.

- **Market Integration of Virtual Power Plants, [4]**
  In this paper a three-stage market model is developed. The model includes the Day-Ahead (Spot) Market, the Intra-Day Market and the Regulating Power Market. The main hypothesis is that the Virtual Power Plant can generate additional profit by trading across several markets. It is found that even though profits do increase as more markets are penetrated, the size of the profit is strongly dependent on the quality of flexibility.

- **Heuristic Optimization for the Discrete Virtual Power Plant Dispatch Problem, [5]**
  In this paper the considered flexible consumers are discrete batch processes, and the associated optimization problem is denoted the Discrete Virtual Power Plant Dispatch Problem. It is proved formally that the Discrete Virtual Power Plant Dispatch Problem is NP-complete. Next tailored versions of the heuristic algorithms *Hill Climber* and *Greedy Randomized Adaptive Search Procedure* (*GRASP*) are developed. The algorithms are tuned and tested on portfolios of varying sizes. By far the best results are obtained by the method *GRASP Sorted*. This method can determine solutions, which are both agile (sorted) and have very little slack even for problems of 100.000 units and 100 samples with a computation time of just 10 seconds.

Notice that throughout the thesis certain terms are used interchangeably depending on the context e.g. assets/facilities/systems/units and balancing/dispatch/scheduling/portfolio coordination.

2 | Background

This section provides a background and state-of-the-art overview for Research Question 1 to 3 proposed in Section 1.2.

As discussed earlier this Thesis makes the assumptions that the Smart Grid must eventually become self-financing and also assumes that Smart Grid players must develop their business within the existing frameworks of deregulated electricity markets. Section 2.1 therefore gives an introduction to the Nordic electricity markets, as this introduction will be used to investigate Research Question 1 in Section 3.2.

What has so far been denoted portfolio coordination is more generally referred to as demand side management. Demand side management falls into the main categories of indirect and direct control. In this Thesis the control strategy discussed in Section 3.2 falls closest to indirect control, whereas the control strategy considered in Section 3.3 to 3.7 falls in the direct control category. Background on indirect and direct control will therefore be reviewed in Section 2.2.

In Section we will suggest the *Buckets*, *Batteries and Bakeries* taxonomy for modelling flexibility, so background on flexibility modelling is given in Section 2.3. Finally Section 2.4 discusses optimization in Smart Grid literature.

2.1 Nordic Electricity Markets

In the Nordic countries the balance between production and consumption at the market level is maintained by means of Day-Ahead Markets, Intra-Day Markets, Regulating Power Markets and Balancing Power Markets (after-day settlement) [22], see Figure 2.1.

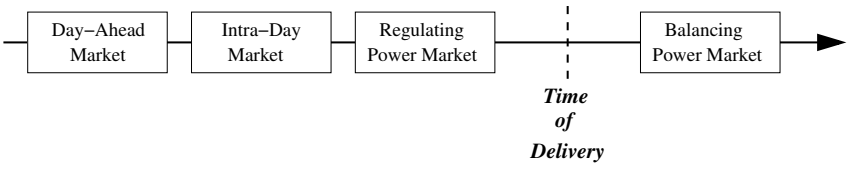

Figure 2.1: Timeline of Nordic electricity markets.

As the name suggests, the Day-Ahead Market (the Spot Market) operates before the actual time of delivery. Producers and wholesalers make bids for production and con-

sumption in future time slots and prices are settled based on a double auction. Once prices on the Day-Ahead Market are settled (Market Clearing), the market is closed.

On the Day-Ahead Market producers and wholesalers have made bids based on the best available knowledge at the time of bidding. As time progresses, however, better forecasts become available. The Day-Ahead Market is therefore followed by the Intra-Day Market (the Elbas Market), where players have the option of adjusting their initial production and consumption schedules in future time slots. The Intra-Day Market is a continuous market where trading takes place up until one hour before the hour of delivery. The Intra-Day Market consists of two lists, which are continuously updated: One list for power purchases and one for power sales. Whenever there is a match within these lists (meaning that a player is willing to purchase power at a price, which is higher than another players sales price), these two bids are activated and removed from the lists. This means that the Intra-Day market is more bilateral in nature than the other markets.

If players do not follow the schedule generated on the Day-Ahead and Intra-Day markets, they generate a need for balancing, i.e. up- or down-regulation. Up- and down-regulation are performed by spare capacity denoted *reserves*, [23]. Traditionally, reserves are provided by specific power plants, which are operating at less than full capacity, so they can ramp up or down as needed.

In the Nordic markets reserve services are traded on the Regulating Power Market. Having a designated power market ensures that a competitive price is paid for Regulating Power. In the Regulating Power Market, bids can be made up to 15 minutes before the hour of delivery. If a need for regulation arises during the hour of operation, then bids are activated in accordance with the highest price of the block of most inexpensive bids until the requested regulation is accumulated.

Since reserves are the 'back-up plan' of the power system it is important to insure sufficient capacity is available in the Regulating Power Market. Therefore certain suppliers of reserves are given an availability/reservation payment for each hour of the day (price/MW/Hour). By accepting this availability payment suppliers commit to supplying a certain quantity to the Regulating Power Market during a specific hour. Thus, on the Regulating Power Market there are both suppliers who have and have not received a reservation payment, [24].

After the actual time of delivery, metered data of actual production/consumption is evaluated. In the after-day settlement (or Balancing Power Market), producers and wholesalers are invoiced according to their trades across the Day-Ahead, Intra-Day and Regulating Power Markets. In the Balancing Power Market the cost of Regulating Power is therefore transferred to any players who have not provided what they have committed to in the Day-Ahead and Intra-Day Market.

2.2 Indirect and Direct Control

Indirect Control

Indirect control is also denoted price signalling. This is because indirect control consist of transmitting a price signal to flexible consumers in order to incentivise a certain behaviour. Flexible units are, however, not obliged contractually to respond to the price signal. The main benefits of indirect control is the simplicity and unit autonomy provided by the set-up. Simplicity follows from, that prices have to be transmitted the consumer, but two

way communication is not required. Settlement can simply be done off-line based on measurements of consumption. The autonomy of flexible units is also well preserved as they can simple choose not to respond to the given incentives. Also, flexible units are not required to disclose any private information about system characteristics or state. A conceptual analysis of indirect control is given in [25] and examples of demand side management via price signalling are given in [26], [27] and [28].

An argument against price signalling is that some consumers might be incapable of identifying their own demand curve (i.e., instantaneous quantity responsiveness as a function of real-time price). This can happen when their objective is to receive a service, which is a function of energy use over time rather than instantaneous consumption, [29]. In [28], however, it is demonstrated that a flexible consumer can reduce electricity cost by approximately 7% by basing consumption decisions on instantaneous and past prices only. In [28] a so-called relative price is computed from recent and current prices. This means that the consumer perception of whether prices are high or low is based on recent price levels and fluctuations, but no price projections are provided. Through an experimental set-up, which includes a micro-CHP unit (Combined Heat and Power) and a space heating system, it is verified that the price responsive consumer can reduce electricity cost by the aforementioned 7%.

A further argument against price signalling is the issue of stability. Stability concerns of indirect control are investigated in [30], where it is assumed that consumers are given an estimated price signal spanning a future horizon. Two kinds of consumers are investigated, namely solely-price-optimized consumers and comfort consumers. Based only on a price minimizing objective the solely-price-optimized consumers construct their consumption schedule. The comfort consumers on the other hand also include a discomfort term to their objective, specifically a cost of too high or low indoor temperature levels. Based on this setup it is demonstrated that if more than two solely-price-optimized consumers are considered then the system can become unstable. Here unstable means that prices and consumers behaviour do not converge over time. It is also demonstrated that if the discomfort term is large enough then stability can be guaranteed for arbitrarily large populations of comfort consumers.

Direct Control

The alternative to indirect control is direct control. In direct control a nonprice-based signal such as temperature setpoint or direct dictation of power consumption is used to control flexible units. The control signal is provided by a third party and there exists a contractual agreement between third party operator and flexible units. Flexible units are therefore contractually obliged to follow the provided reference as long as the utilization of the flexible system stays within the bounds of the contract. An information modelling architecture for direct control of flexible units is presented in [31].

In [32] a three level hierarchical direct control set-up is developed for a portfolio of *Bucket* type consumers. The hierarchical structure is designed to facilitate plug-and-play control and remain stable for an increasing number of units. It is demonstrated though simulations that the proposed architecture can supply reference tracking by distributing imbalances to the flexible units.

Direct control based on more complex consumers models is investigated in [33] and [34]. Two systems are considered, namely a supermarket refrigeration system and a

chiller with ice storage. Both papers investigate how to supply downward regulating power from the two-system portfolio. In [33] it is demonstrated that with the direct control set-up, the aggregator will be better able to follow the considered power reference than what can be achieved with a proposed indirect control set-up. In [34] it is concluded that under the direct control framework the power reference level (low, medium, high) determines how power should be distribute between the refrigeration system and the ice storage. This demonstrates that when considering a heterogeneous portfolio under a direct control policy a better utilization of flexibility can be obtained by coordinating the portfolio than what could be achieved by considering units individually.

Paper [35] looks into the problem of managing a large number of thermostat-based appliances with on/off operation. First it is concluded that the memory and computation time requirements associated with a centralized MPC coordination strategy renders that approach impossible. A distributed control structure is therefore developed based on an aggregated consumer response model and a single global coordination signal. It is demonstrated that the proposed architecture can efficiently coordinate the consumption of 1000 thermostat-based devices. This efficiency is however derived from the aggregated model of consumer responsiveness of the relatively homogeneous units under consideration. It could therefore be argued, that such an aggregated responsiveness model for heterogeneous units would be far more complex and possibly less accurate.

2.3 Flexibility

As discussed earlier flexibility is central to the Smart Grid discussion, but a formal definition of flexibility is difficult to give. We have therefore suggested the *Buckets*, *Batteries and Bakeries* taxonomy as way of formalizing flexibility. The taxonomy focuses on the constraints of

1. Power Capacity,
2. Energy Capacity,
3. Energy level at a specific deadline, and
4. Runtime,

since these are widely found in physical systems. Based on these constraints three archetypal flexibility models are defined. The archetypes are denoted *Buckets*, *Batteries and Bakeries* and they are presented in Section 3.1. A review of flexibility modelling in Smart Grid literature reveals that the generic models of *Buckets*, *Batteries* and *Bakeries* are not in themselves novel concepts. Several works have been identified (see Table 2.1), which model flexibility in ways very similar to the *Buckets*, *Batteries and Bakeries* taxonomy. Most existing literature, however, focuses on optimized operation of one particular technology. Two examples of papers which also seek to formalize the flexibility concept are however [43] and [36].

In [43] a modelling framework denoted Smart Finite State Devises are presented. Four types of units are included in the framework, namely optional loads, deferrable loads, controllable loads and storage devices. Units are described as Markov Decision Processes where the switch between states are stochastic processes [55]. The approach is similar to

Reference	[36]	[37]	[38]	[39]	[40]	[41]	[42]	[43]	[44]	[45]
Bucket	x	x	x	x	x	x	x	(x)		
Battery					(x)	x		(x)	x	x
Bakery							(x)	(x)		

Reference	[46]	[47]	[48]	[49]	[50]	[51]	[52]	[53]	[54]
Bucket									
Battery	x	x	x	x	x	x	x		
Bakery						x	x	x	x

Table 2.1: Review of flexibility modelling in Smart Grid literature.

the *Buckets*, *Batteries and Bakeries* framework in the sense that the description of flexible consumers is highly abstracted and therefore allows for a categorization of diverse types of units. A weakness of the presented technique is that since the modelling is based on probability it does not guarantee upper and lower bounds in consumer discomfort such as waiting time or temperature levels.

The so-called Power Nodes Framework suggested in [36] is in a sense an even more general setup than both the *Buckets*, *Batteries and Bakeries* taxonomy and Smart Finite State Devises, since a single generic model is suggested to describe anything from storage units and thermal loads to wind farms and conventional generation. The model includes power constraints, ramp-rate constraints, efficiencies and storage capacities. The authors argue that these constraints are included because the grid-relevant aspects of units should be captured by the model while technology-dependent and physical unit properties should be abstracted from. They also note that apart from the constraints included there may be additional ones imposed on the variables, e.g. in order to define certain standard unit types with characteristic properties. It is therefore difficult to determine the boundaries of the framework: Where do we draw the distinction between an extended power node and a conventional system model? And, is e.g. a minimum runtime constraint grid-relevant and should thus be included or is it technology-dependent and should thus be abstracted from?

2.4 Optimization in Smart Grid Literature

When formulating an optimization problem for further research within the Smart Grid area how to model flexibility and whether to investigate direct or indirect control are not the only central choices. There are also several relevant objectives, which can be included in the problem formulation, such as

- consumption Scheduling, [53],
- grid congestion management, [56],
- minimization of generation cost, [57],
- minimization of user discomfort, [62], and

Reference	Simulation samples	Num. units	Comp. time
[53]	144	<20	1.2 sec
[56]	24	600	30 sec
[57]	24	5	50 sec
[58]	Unclear	20858	1.1 min
[59]	288	6	3.7 hours
[60]	24	3504	6 hours
[61]	24	50	83 hours

Table 2.2: Overview of computation times reported in recent scientific publications on optimization in Smart Grid applications. Simulation samples is the number of discrete time steps that the considered simulation horizon has been split into, [5].

- management of reactive power and voltage control, [58].

Depending on the objectives included the optimization problem can again be formulated as

- mixed integer linear programming, [53],
- non-linear optimization, [58],
- quadratic optimization, [61], and
- stochastic optimization, [62].

Based on the type problem formulation countless methods and solution techniques are the available within the vast area of optimization. A review of computation times for optimization in Smart Grid literature, however, reveals that computation time is still a very real challenge. Table 2.2 summarizes problem size and computation times for recent scientific publications related to Smart Grid optimization. Obviously computation times are highly dependent on the specific structure of the considered problem and the software and platform used for calculation. Nonetheless, Table 2.2 does give the general impression that computation times are still quite a lot longer than what one can expect to be acceptable for a fully deployed Smart Grid operating in real time. This is the case even though several of the cited references investigate heuristic rather than exact optimization methods.

In this Thesis we will investigate consumption scheduling formulated as a mixed integer linear program. This has also been investigated in [53] and [57].

In [53] indirect control of *Bakery* type consumers are considered. The formulated optimization problem is solved using the commercial software package CPLEX, [65]. This software package offers the functionality of terminating as soon as a feasible solution is found and this functionality is thoroughly investigated in [53]. Through simulations it is found, that when considering only one unit, the difference between the optimal solution and terminating as soon as a feasible solution is found is only 1% in terms of solution quality. The difference in computation time between the two approaches is not clear from

the context. It is also found, however, that the first feasible solution method fails for more than 20 units due to lack of memory.

In [57] the mixed integer program is formulated using fuzzy set logic and solved using a Genetic Algorithm and Evolutionary Particle Swarm Optimization. The algorithms are tested on an experimental set-up, which includes a photovoltaic panel, a wind turbine, a fuel cell, light bulbs and a storage device. It is found that the Genetic Algorithm and Evolutionary Particle Swarm Optimization can obtain comparable results in terms of solution quality and computation time, but the Genetic Algorithm requires nearly twice as much memory as Evolutionary Particle Swarm Optimization.

3 | Summary of Contributions

This section provides responses to the Research Questions 1 to 3 formulated in Section 1.2. Responses are based on findings from papers [1] to [5] and these papers are referenced throughout. This Chapter does not give consecutive summaries of papers, but instead contents is summarized in the following main contributions:

1. Introduction of taxonomy for modelling of flexibility, [3], [4],
2. Demonstration of that better quality flexibility is more valuable, [4],
3. Introduction of the concept of agility and demonstration of that agility can create value, [1], [2], [3],
4. Demonstration of and proof that computational complexity of portfolio coordination can become critical for the Virtual Power Plant and the development of heuristic algorithms to mitigate the challenge, [3], [5],

These four subjects will be addressed in Section 3.1, 3.2, 3.3 and 3.4 to 3.7 respectively.

After the introduction of the *Buckets*, *Batteries and Bakeries* taxonomy in Section 3.1 several different concepts and methods are presented. Most of these will be illustrated by examples or simulations. Each concept or method is illustrated with the most appropriate portfolio, which means that the considered portfolio will change several times throughout the Chapter. To assist the reader light blue is used for *Buckets*, medium blue for *Batteries* and dark blue for *Bakeries* (See Figure 3.1) throughout. Furthermore Table 3.1 gives an overview of the portfolios considered in each section.

In Sections 3.4 to 3.7 the computational complexity of portfolio coordination is investigated. In Section 3.5 the term exact method is used to oppose heuristic optimization methods. By exact method we therefore denote an optimization technique, which will determine an optimal solution of a given problem if provided with sufficient computation time.

3.1 Flexibility

To investigate the research questions posed in Section 1.2 a first critical choice is how to model the flexibility of the systems in the Virtual Power Plant portfolio. In this thesis this is done by defining a taxonomy to express flexibility in a consistent manner. The taxonomy is denoted *Buckets*, *Batteries and Bakeries* and it focuses on the constraints of

Section	Subsection	Page	Portfolio
3.1		17	*A Bucket*, *a Battery* and *a Bakery* as depicted in Figures 3.2, 3.3 and 3.4.
3.2		23	*A Bucket*, *a Battery* and *a Bakery* with parameter values $\underline{P} = -1MW, \overline{P} = 1MW, \underline{E} = 0MWh, \overline{E} = 3MWh$. Drain of 10% on *Bucket*.
3.3	*Agility and Prediction Errors*	28	Three *Bakeries* with parameter values as depicted in Figure 3.10.
3.3	*Agility and Length of Prediction Horizon*	30	Nine *Buckets* having $\overline{P} = \overline{E} = 0$ and additional parameters as given in Table 3.4.
3.5	*CPLEX*	37	First 50 *Bakeries* then 100 *Bakeries*.
3.5	*Dynamic Programming*	37	Portfolios of 10 to 15 *Bakeries*.
3.5	*Dantzig-Wolfe Decomposition*	41	Four *Batteries* with parameter values $\overline{P} = 2$ and $\overline{E} = 2$. After that four *Bakeries* with identical parameter values.
3.7	*Hill Climber and GRASP*	48	Portfolios of 1.000, 10.000 and 100.000 *Bakeries*.
3.7	*Prediction and Agility*	54	Portfolios of 5 *Buckets*, 50 *Batteries* and 50 *Bakeries*.

Table 3.1: Overview of portfolios considered in each section of Chapter 3.

1. Power Capacity,
2. Energy Capacity,
3. Energy level at a specific deadline, and
4. Runtime,

since these are widely found in physical systems. Obviously, this is by no means an exhaustive definition, since there are numerous other constraints, parameters and dynamics, which could also be considered (See Paper [1] for further discussion). In fact, almost arbitrarily detailed modelling of any real world system can be imagined. However, if highly customized models are used at the lowest level as the Smart Grid is deployed, then algorithms and interfaces at higher levels must also be able to handle such arbitrarily complex models. In practice this would be very challenging and costly, so the taxonomy should be seen as an attempt at simplification and standardization.

Taxonomy

The Bucket, *The Battery* and *The Bakery* are three simple flexibility models, which are constructed based on the constraints 1) to 4).

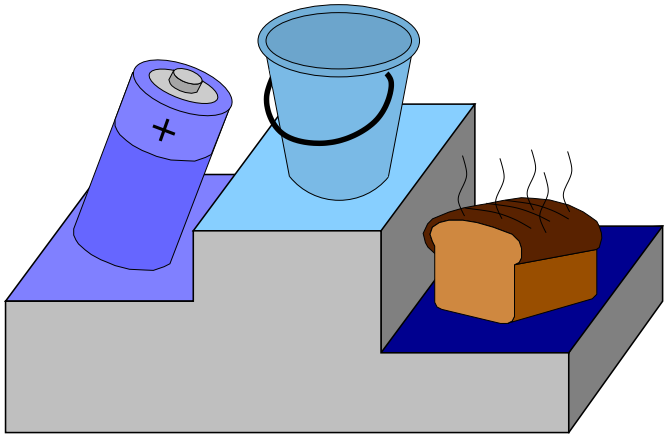

Figure 3.1: *Buckets*, *Batteries and Bakeries* is a taxonomy for modelling flexibility in Smart Grids. Light blue, medium blue and dark blue will be associated with *Buckets*, *Batteries and Bakeries* respectively throughout this section.

The suggested framework is a proper taxonomy in the sense that there is a hierarchical relationship between the three models. This means that *a Bucket* provides a better quality of flexibility than *a Battery*, which is again superior to *a Bakery* (see Figure 3.1). Here, better quality means less constrained, not necessarily more flexible or valuable. This distinction must be made because the flexibility of a system is not just determined by constraints, but also by the specific parameter values of the system. That is, a large *Battery* could in some situations be considered ”more flexible” or ”more valuable” than a small *Bucket*, even though the *Bucket* is a better quality flexibility than the *Battery*.

Formal definitions of a *Bucket*, a *Battery* and a *Bakery* are given in Definition 1, 2 and 3 below. T_s denotes the size of the time step, $\underline{P}$ and $\overline{P}$ denote limits on consumption rate, $\underline{E}$ and $\overline{E}$ denote limits on energy storage levels and $v(k)$ is a boolean-valued variable stating whether or not a *Bakery* is running at sample k.

The first model in the taxonomy is denoted *the Bucket* and it is a power and energy constrained integrator. The *Bucket* could for example be a simplified model of a house with a heat pump, which is used for energy storage. The average consumption of the heat pump should then be considered as a fixed load, and the *Bucket* seen as only the controllable consumption around that average. Thus, the ”negative energy” illustrated in Figure 3.2 indicates that the thermal energy stored in the house is below the average for that time of day. The *Bucket* is defined by

Definition 1 (*Bucket*)**.** The dynamics and constraints of a *Bucket* are

$$\textit{Bucket}(k)\colon E(k+1) = E(k) + T_s P(k) \tag{3.1}$$
$$\underline{P} \leq P(k) \leq \overline{P} \tag{3.2}$$
$$\underline{E} \leq E(k) \leq \overline{E} \tag{3.3}$$
$$E(0) = E_0, \tag{3.4}$$

where $k = 0, 1, \ldots, \infty$, $\underline{P} \leq 0 \leq \overline{P}$ and $\underline{E} \leq E_0 \leq \overline{E}$.

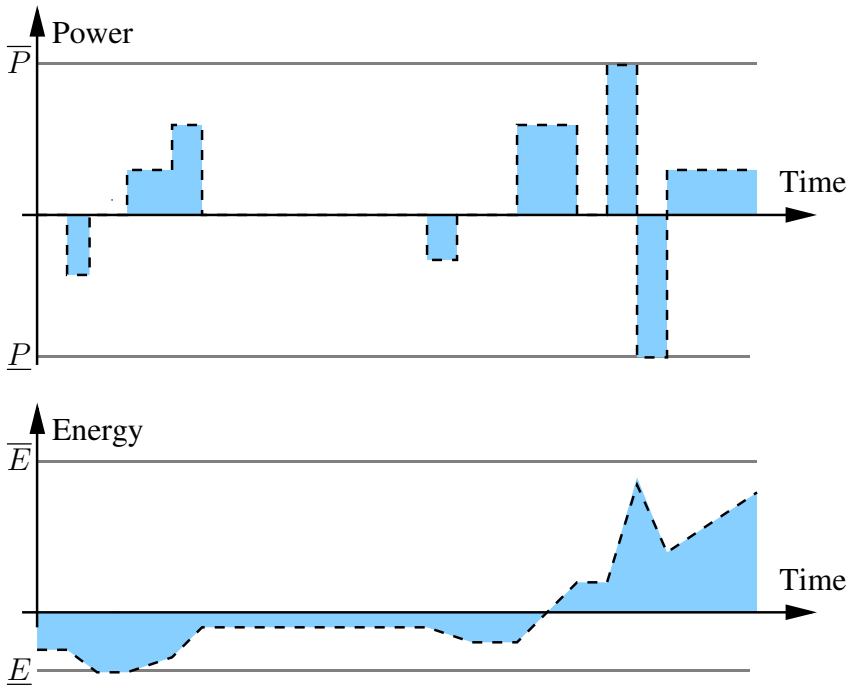

Figure 3.2: Example of a consumption profile for a *Bucket*. The *Bucket* is equivalent to a power and energy constrained integrator.

Like the *Bucket*, the *Battery* is a power and energy constrained integrator, but with the added restriction that the unit must be fully charged at a specific deadline. The *Battery* could be emulating an electric vehicle, which must be ready for operation at a specific time. The *Battery* is defined by

Definition 2 (*Battery*)**.** The dynamics and constraints of a *Battery* are

$$\textit{Battery}(k)\colon E(k+1) = E(k) + T_s P(k) \tag{3.5}$$
$$0 \leq P(k) \leq \overline{P} \tag{3.6}$$
$$0 \leq E(k) \leq \overline{E} \tag{3.7}$$
$$E(0) = E_0, \tag{3.8}$$
$$E(K_{end}) = \overline{E}, \tag{3.9}$$

where $k = 0, 1, \ldots, \infty$, $K_{end} \in \mathbb{N}$, $0 \leq \overline{P}$ and $0 \leq \overline{E}$.

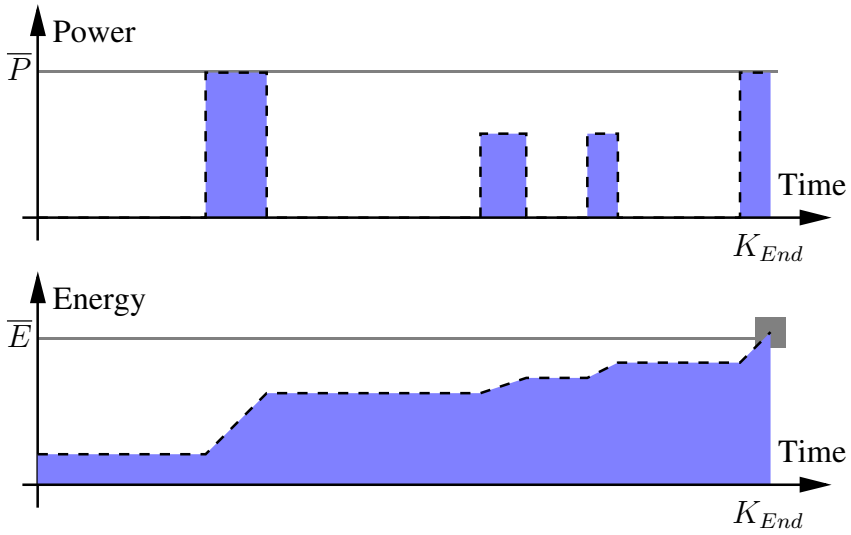

Figure 3.3: Example of a consumption profile for a *Battery*. The *Battery* is a power and energy constrained integrator, which must be "charged" to level $\overline{E}$ by time K_{end}.

Finally the *Bakery* extends the *Battery* with the additional constraint that the process must run as one continuous batch and at constant power. The *Bakery* could be a commercial green house, where plants must receive a specific amount of light each day. This light must, however, be delivered continuously and at a constant level to stimulate the photosynthesis of the plants. A similar requirement exists for baking bread: To achieve a nice loaf of bread we must bake for thirty minutes at approximately 200°C. Baking for two hours at 50°C, however, will just leave us with warm dough. Hence the name *Bakery* for the batch model. The *Bakery* is defined by

Definition 3 (*Bakery*)**.** The dynamics and constraints of a *Bakery* are

$$\textit{Bakery}(k)\colon E(k+1) = E(k) + T_s P(k), \tag{3.10}$$
$$P(k) = \overline{P} v(k) \tag{3.11}$$
$$0 \leq E(k) \leq \overline{E}, \tag{3.12}$$
$$E(0) = E_0, \tag{3.13}$$
$$E(K_{end}) = \overline{E}, \tag{3.14}$$
$$0 \leq \sum_{l=k}^{k+K_{run}-1} v(l) - K_{run}\Big(v(k) - v(k-1)\Big), \tag{3.15}$$

where $k = 0, 1, \ldots, \infty$, $0 \leq \overline{P}$, $\overline{E} = \overline{P} K_{run}$, $v(k) \in \{0, 1\}$, $K_{end} \in \mathbb{N}$ and $K_{run} \in \mathbb{N}$. Here, inequality (3.15) is the minimum runtime constraint, which ensures that if $v(k) - v(k-1)$ is one, then $v(l), l = k+1, k+2, \ldots, K_{Run} - 1$ must also be one; that is, once the *Bakery* is activated, it must complete its consumption immediately and

continuously. Obviously more complicated power sequences than constant consumption can be imagined, but the main point is that the only flexibility is in the activation time.

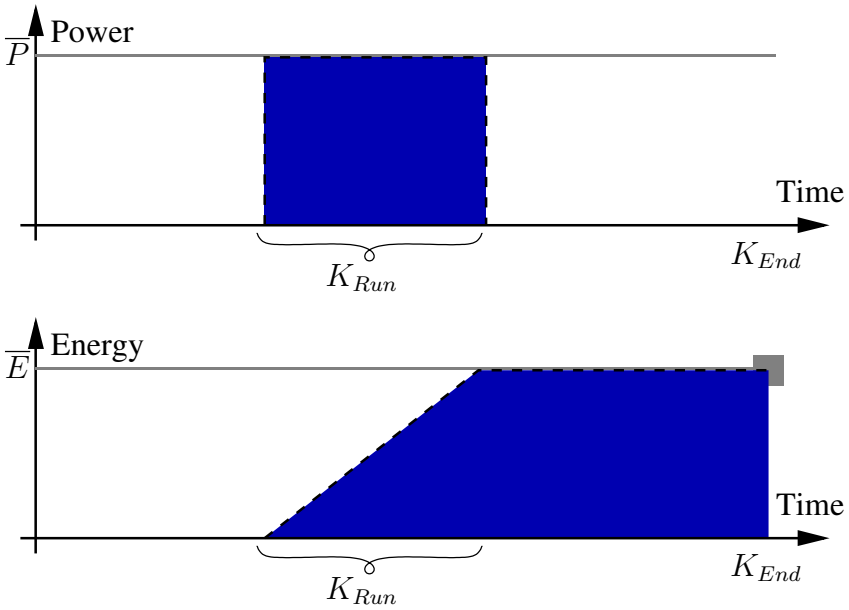

Figure 3.4: Example of a consumption profile for a *Bakery*. The *Bakery* is a batch process, which must be finished by time K_{end}. The process has constant power consumption and the run time is K_{run}.

In the remainder of the thesis a flexible consumer will also be denoted a local unit. A portfolio of N local units of the type *Buckets*, *Batteries and Bakeries* will then be denoted $\{LU_i\}_{i=1,2,...,N}$. The aggregated consumption of a portfolio consisting of the *Bucket*, the *Battery* and the *Bakery* depicted in Figure 3.2, 3.3 and 3.4 respectively is given by the profile in Figure 3.5.

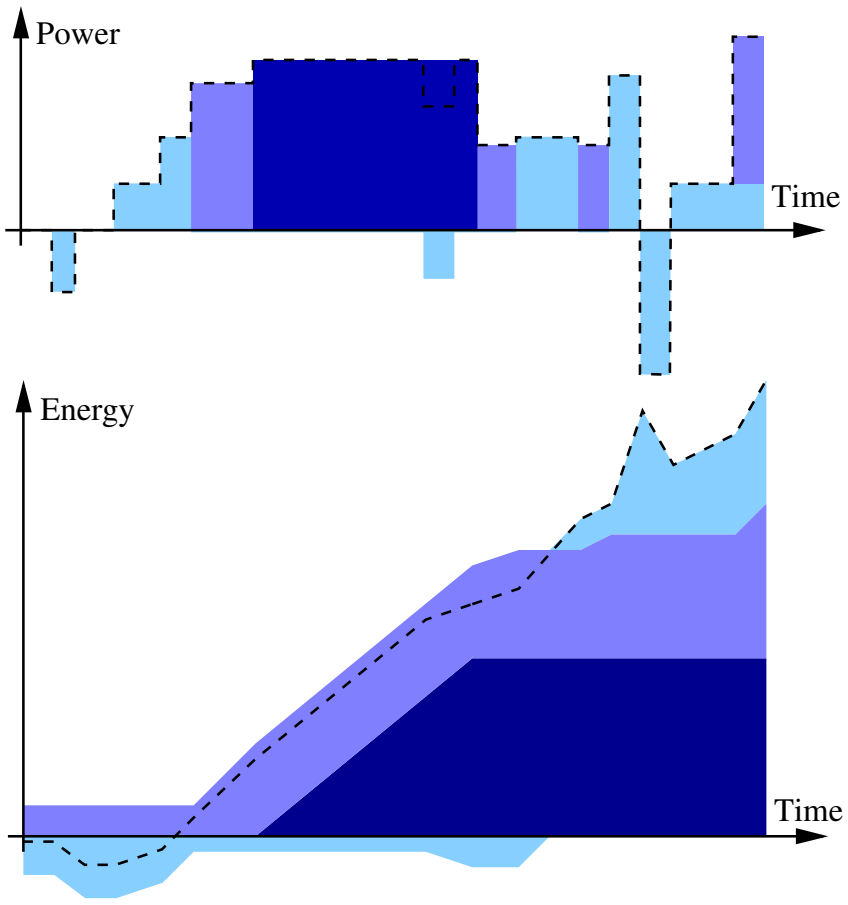

Figure 3.5: The dashed line corresponds to the aggregated consumption for the *Bucket*, *Battery* and *Bakery* depicted in Figures 3.2, 3.3 and 3.4 respectively.

3.2 Value of Flexibility

In this section the taxonomy will be used to test the hypothesis that different quality of flexibility have different revenue potential. This was investigated in [4] where it was attempted to assign monetary value to different flexibility qualities. The hypothesis is tested in a simple set-up, which is intentionally not extended with assumptions and correlations, which might and might not turn out to hold true for a fully deployed Smart Grid. There-

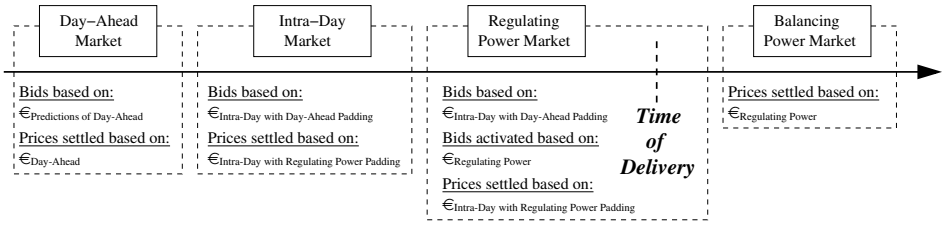

Figure 3.6: Timeline of electricity market model.

fore the market model was developed to include historic prices only. Since the model is based on historic data there is no feedback in the formation of prices, meaning that prices are not generated dynamically. Consequently, the model is only valid under the assumption that the total amount of flexibility bid into the system is small enough not to affect the formation of prices significantly.

The considered portfolio consists of one unit of each type in the taxonomy and the same parameter values are used for all three units. This means that units are not modelling the flexibility of any specific systems, but instead it is possible to directly compare the effects that different constraints have on the revenue potential. Notice that a drain of 10% has been added to the *Bucket* model meaning that 10% of the energy stored in the model (that is $E(k)$ in Equation (3.1)) drains from the *Bucket* with each sample/hour.

Model of Electricity Markets

To perform calculations a three-stage market model (see Figure 3.6) is developed based on the market description in Section 2.1. The model consists of a series of optimization problems, which the Virtual Power Plant solves one by one to determine how to bid into the different markets. With each optimization problem the latest and most updated information is used. All optimization problems are formulated explicitly in Paper [4], but here we will just give a short summary and state the main results.

The first stage of the market model is the Day-Ahead Market, which the Virtual Power Plant can bid into based on predictions of market prices (in Paper [4] sensitivity to prediction quality is also investigated). Remember that the portfolio consist of a *Bucket*, a *Battery* and a *Bakery*, so the Virtual Power Plant must purchase the baseline power required to charge the *Battery* and run the *Bakery*. The Virtual Power Plant can also start to gain a profit from the *Bucket* if there is a step up in prices from one hour to the next. The Virtual Power Plant can then purchase power for the *Bucket* at the lower price in the first hour and sell the remaining 90% at the higher price of the next hour to make a profit.

In the second stage of the model the Intra-Day market is opened. If there is activity on the Intra-Day market, the Virtual Power Plant can do additional trading here to further increase its profit. This means that the Virtual Power Plant has the option of selling power bought on the Day-Ahead Market and change the consumption schedule for each local unit if this can increase earnings.

In the third stage of bidding the Virtual Power Plant must make up- and down-regulation bids into the Regulating Power Market. This is done one hour at a time and again the Virtual Power Plant must solve a specific optimization problem to determine the

price, which should be bid into the market. Price-wise up-regulation bids should be as low as possible to get activated and down-regulation bids should be as high as possible to get activated. Still, however, the Virtual Power Plant should of course only bid at prices, which it projects will make a profit.

Finally, there is an independent stage for the Balancing Market, where the appropriate imbalances are paid/compensated at the price of Regulating Power.

Results of Electricity Market Trading

The analysis focuses on the Danish electricity market, so Day-Ahead prices, average Intra-Day prices and Regulating Power prices from price zones DK1 and DK2 (see Figure 3.7) in 2010, 2011 and 2012 are used in calculations. The data can be downloaded from [63]. Portfolio parameters are $\underline{P}$ = -1 MW, $\overline{P}$ = 1 MW, $\underline{E}$ = 0 MWh, $\overline{E}$ = 3 MWh for all three units, which corresponds to a run time of three hours for the *Bakery*, that is $K_{run} = 3\ hours$.

Figure 3.7: The price zones of western and eastern Denmark are referred to as DK1 and DK2 respectively.

Simulations show that for the *Bucket* it is possible for the Virtual Power Plant to achieve a profit in all three considered years in both DK1 and DK2, see Table 3.2. For the *Battery* and *Bakery*, however, the base load requirements mean that the Virtual Power Plant cannot make a profit on these two types of units, see Table 3.3. It is, however, possible for the Virtual Power Plant to achieve considerable savings for the *Battery* and *Bakery* in the second and third market stages after the initial purchase of base load power in the Day-Ahead Market. For the *Battery* these savings are in the order of 3% to 24% and for the *Bakery* between 1% and 10% depending on the year and price zone.

A sensible contractual agreement between asset owner and Virtual Power Plant could thus be that the asset owner should pay the expense of purchasing base load power in the Day-Ahead Market. Any additional profit gained in the Intra-Day and Regulating Power Markets should then be shared evenly between asset owner and Virtual Power Plant. With this setup, Figure 3.8 shows which flexibility type is most profitable for the Virtual Power

Year	Zone	***Bucket*** Day-Ahead	Intra-Day	Regulating
2010	DK1	-6.823	-7.544	-20.792
	DK2	-21.409	-21.998	-28.937
2011	DK1	-8.655	-9.969	-27.880
	DK2	-10.156	-11.386	-30.047
2012	DK1	-11.707	-14.639	-35.981
	DK2	-14.541	-17.383	-39.282
Total		-73.292	-82.919	-182.919

Table 3.2: Profit (Negative numbers) in € obtained by trading the *Bucket* according to the developed model.

Year	Zone	Scenario	***Battery*** Day-Ahead	Intra-Day	Regulating	***Bakery*** Day-Ahead	Intra-Day	Regulating
2010	DK1	Day,	50.984	51.281	48.877	51.307	51.432	50.793
		Night	38.154	38.047	35.083	38.233	38.095	35.605
	DK2	Day	62.139	62.367	58.965	62.453	62.872	61.609
		Night	44.067	43.927	42.775	44.170	43.935	43.329
2011	DK1	Day,	53.178	52.488	47.403	53.404	52.440	50.425
		Night	37.376	36.873	34.103	37.463	37.041	35.686
	DK2	Day	55.192	55.001	47.221	55.460	55.786	51.084
		Night	37.539	37.449	35.430	37.621	37.403	36.638
2012	DK1	Day,	39.843	39.345	31.409	40.059	40.058	37.176
		Night	26.638	26.296	24.587	26.682	26.531	25.895
	DK2	Day	41.171	40.186	31.188	41.420	40.757	37.393
		Night	26.724	26.317	24.716	26.752	26.449	26.048
Total			513.005	509.576	461.757	515.025	512.799	491.681

Table 3.3: Costs in € accumulated by trading the *Battery* and *Bakery* according to the developed model. The first market stage is the Day-Ahead Market, the second market stage is the Intra-Day Market and the third market stage is the Regulating Power Market. Notice that considerable savings are obtained in the second and third market stages after the initial purchase of base load power in the Day-Ahead Market.

Plant. It is found that the *Bucket* is far more profitable than the *Battery*, which again generates much larger profits than the *Bakery*. This is the case for both DK1 and DK2 in 2010, 2011 and 2012. If total profits for DK1 and DK2 and all three considered years are summed up the *Battery* earns 14% and the *Bakery* only 6% of the profit earned by the *Bucket*. This confirms the hypothesis that a better quality of flexibility is also more valuable.

Paper [4] also gives a detailed Single-Day illustration of how the three flexibility qualities should have been traded on February 3^{rd}, 2012 according to the model. This example also clearly establishes how different types of constraints prompt that units can be offered to the markets at different times. Consequently the revenue potential depends strongly on the quality of flexibility.

Additional contributions of Paper [4] are the results that the Virtual Power Plant can increase its profit by trading in several markets and that bidding into several markets also

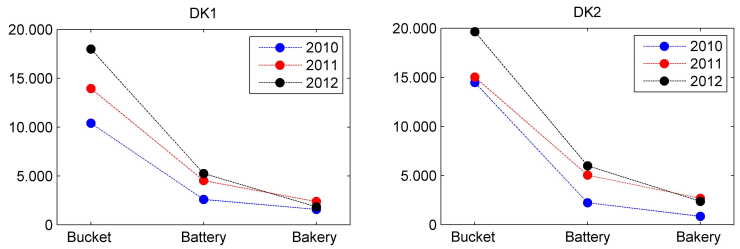

Figure 3.8: Estimated Virtual Power Plant profit in € for each type of flexibility in DK1 and DK2 respectively.

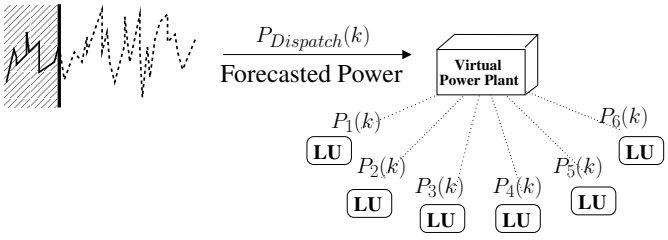

Figure 3.9: The Virtual Power Plant Dispatch Problem.

makes profits surprisingly robust to errors in Day-Ahead price predictions.

3.3 Agility in Dispatch

Since it has been demonstrated that better quality of flexibility is also more valuable the idea of agility is developed. The term agility means to maximize the quality of flexibility in the portfolio as flexible units are dispatched. This section will first introduce the Virtual Power Plant Dispatch Problem and then illustrate the benefits of agility in different unfavourable situations.

To formulate the Virtual Power Plant Dispatch Problem, see Figure 3.9, let $P_{Dispatch}(k), k = 1, 2, \cdots, K$ denote a fluctuating power supply, which must be dispatched to the portfolio. As discussed in Section 1.1 depending on the market integration of the Virtual Power Plant the signal $P_{Dispatch}$ could be generated by a master controller, a higher level Virtual Power Plant or be the result of direct market trading (See Figure 1.3). Remember that a portfolio of N local units of the type *Buckets*, *Batteries and Bakeries* is denoted $\{LU_i\}_{i=1,2,\ldots,N}$. At sample k also let $P_i(k)$ denote the power dispatched to unit i, and any portion of $P_{Dispatch}(k)$ which cannot be dispatched to the portfolio is denoted $\mathcal{S}(k)$.

The objective is to minimize the residual power, that is $|\mathcal{S}|$, so the Virtual Power Plant Dispatch Problem can be formulated as

$$f(P_i(\cdot)) = \min_{P_i(\cdot)} \sum_{k=1}^{K} |\mathcal{S}(k)| \tag{3.16}$$

$$s.t.$$

$$\sum_{i=1}^{N} P_i(k) + \mathcal{S}(k) = P_{Dispatch}(k), \tag{3.17}$$

and also subject to the dynamics and constraints of $\{LU_i\}_{i=1,2,\ldots,N}$.

The remainder of this section will demonstrate the advantages of adding agility to the standard formulation of the Virtual Power Plant Dispatch Problem. Adding agility to the dispatch problem is really an attempt to maximize the solution space, which is the basis of dispatch problems to be solved in future time steps. There are therefore two situations in which agility is beneficial:

1. when predictions of $P_{Dispach}$ are erroneousness, and

2. when the length of the prediction horizon of $P_{Dispatch}$ is not sufficient.

Firstly the benefit of agility in case of erroneous predictions is demonstrated, which is done by considering a simple dispatch problem for a small portfolio of *Bakeries*. Next results from Papers [1] and [2] are summarized to illustrate the benefit of agility when prediction horizons are not sufficiently long. Here the portfolio is composed of *Buckets* having $\underline{P} = \underline{E} = 0$. Finally it is demonstrated how to build agility into a portfolio of mixed units by use of so-called Agility Factors. This also demonstrates how agility depends not only on the type of flexibility but also on parameter values.

Agility and Prediction Errors

To illustrate the usefulness of agility in case of erroneous predictions, consider a portfolio consisting of *Bakery*$_1$, *Bakery*$_2$ and *Bakery*$_3$ each having a run time of one hour and deadlines of 17:00, 14:00 and 15:00 respectively, see Figure 3.10. Also say that at 12:00 it is predicted that sufficient power to satisfy all three *Bakeries* will be available between the hours of 12:00 and 15:00, see Figure 3.11.

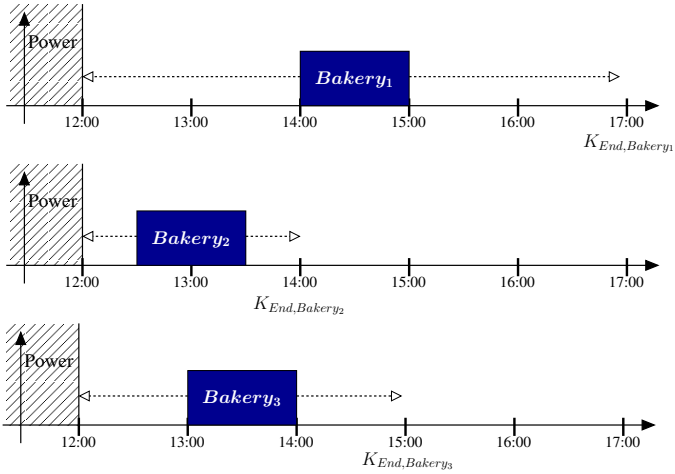

Figure 3.10: Consider a portfolio consisting of $Bakery_1$, $Bakery_2$ and $Bakery_3$ each having a run time of one hour and deadlines of 17:00, 14:00 and 15:00 respectively.

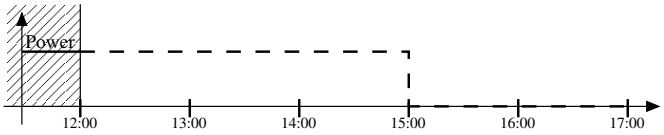

Figure 3.11: Power availability forecasted at 12:00.

Given this setup dispatching in the order $A = \{Bakery_1, Bakery_2, Bakery_3\}$ and in the order $B = \{Bakery_2, Bakery_3, Bakery_1\}$ both yield optimal solutions of the dispatch problem, see Figure 3.12. Now, if the projection of power availability is correct, then it is obviously unimportant to distinguish between dispatch A and B as they are both optimal given that they have zero slack. However, if the power thought to be available between 14:00 and 15:00 is delayed by two hours (see Figure 3.13) then dispatch B can remain optimal by moving the start time for $Bakery_1$ from 14:00 to 16:00. For dispatch A, however, this is not an option, because at 14:00 only $Bakery_3$ remains to be started and this unit has deadline at 15:00.

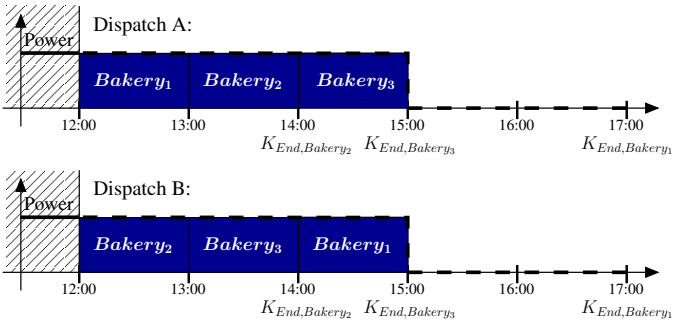

Figure 3.12: If power is available between the hours of 12:00 and 15:00 then dispatch A and dispatch B are both optimal solutions of the considered dispatch problem.

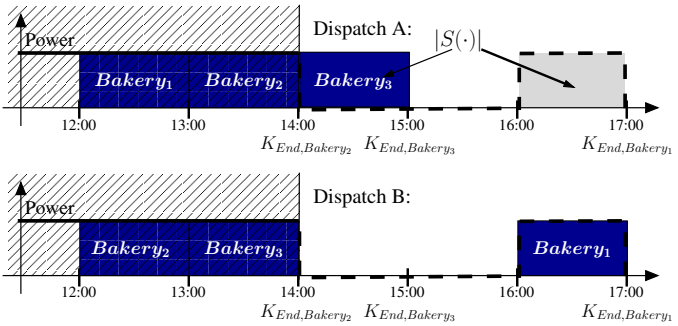

Figure 3.13: As time progresses the agile dispatch B can remain optimal even when the power thought to be available between 14:00 and 15:00 is delayed two hours. Dispatch A cannot do this, since at 14:00 only $Bakery_3$ remains to be started and $Bakery_3$ has deadline at 15:00.

The reason dispatch B can remain optimal is that it is the more agile of the two since *Bakeries* are dispatched in the order of deadlines. This means that dispatch B has left more manoeuvrability for the optimization in later time steps. This illustrates how adding agility to the dispatch problem maximizes the solution space, which is the basis of dispatch problems to be solved in future time steps.

Agility and Length of Prediction Horizon

We first investigated the relationship between agility and prediction horizon in Papers [1] and [2]. Here the portfolio consists of *Buckets* having $\underline{P} = \underline{E} = 0$. The main contribution of Paper [2] is a formal proof that for this type of portfolio there exists an optimal, agile dispatch strategy, which can be found at each sample by solving a quadratic program (see Theorem 1 in Section 3.4 for further details). This dispatch strategy is denoted the agile

strategy and it is strictly non-predictive since at sample k it determines a dispatch based only on the current state of the portfolio and the value of $P_{Dispatch}(k)$.

The agile strategy is juxtaposed to a moving horizon strategy denoted the predictive strategy. The predictive strategy is given perfect prediction of a number of samples of $P_{Dispatch}$. Problem (3.16) to (3.17) is then solved and the results for the first sample is implemented.

Let $P_{Reserve,i}(k)$ denote the upper bound on the power, which can be dispatched to local unit i at sample k. Since only *Buckets* having $\underline{P} = \underline{E} = 0$ are considered it follows that

$$P_{Reserve,i}(k) = \min(\overline{P}_i, \overline{E}_i - E_i(k)),$$

and the maximum power, which can be dispatched to the portfolio at sample k, is

$$P_{Reserve}(k) = \sum_{i=1}^{N} P_{Reserve,i}(k).$$

Since only positive power is considered $P_{Reserve}$ can be used as a measures of the quality of the portfolio. A high value of $P_{Reserve}$ thus means that a better service is provided to the market/master controller.

To compare the agile and predictive dispatch strategies a small simulation example is given. Nine local units are included in the simulations and parameters for these are given in Table 3.4. We set $T_s = 1$ and the predictive strategy is given three samples of perfect prediction of $P_{Dispatch}$.

i	$\overline{P}_i$	$\overline{E}_i$	$E_{i,0}$
1	1	40	0
2	2	50	0
3	3	45	0
4	4	120	0
5	5	175	0
6	6	270	0
7	7	35	0
8	8	160	0
9	9	90	0

Table 3.4: Parameters for the local units.

The simulation results are given in Figure 3.14. It can be seen that after 90 samples the agile and the predictive strategies both run out of flexibility, which is obviously because the two methods have the exact same portfolio at their disposal and have to balance the exact same load. Before that, however, the agile approach does a much better job at preserving $P_{Reserve}$ than the predictive strategy. This illustrates to important points:

1. *Agility can create value:* If the Virtual Power Plant is given not only an activation payment, but also an availability payment (price/reserved quantity/time unit) as explained in Section 2.1, then the higher value of $P_{Reserve}$ will translates directly to higher profit.

2. *Correct predictions do not translate into agility:* The previous example demonstrated that agility could make dispatch more robust of prediction errors. In this example, however, the forecasts, which the predictive strategy is given are actually correct, but clearly this does not in itself translate to agility. The predictive dispatch approach must have a very long (in theory infinite) prediction horizon in order not to violate the agility property.

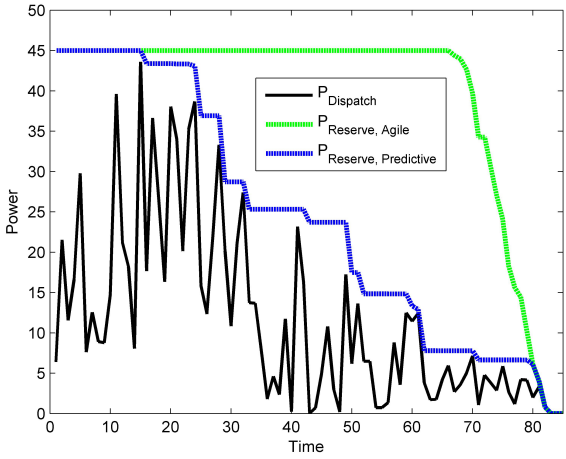

Figure 3.14: $P_{Reserve}$ is an upper bound on the amount of power, which can be dispatched to the portfolio at each sample.
At each sample the predictive dispatch strategy has perfect prediction of $P_{Dispatch}$ over the next three samples; An assumption which is *not* made by the agile strategy.
In the first part of the simulations the two methods perform equally well. After sample 15, however, the agile dispatch strategy is able to compensate for a larger imbalance than the predictive dispatch strategy. This happens eventhough the two methods have the exact same portfolio at their disposal and have to balance the exact same load.

Agility Factors for Mixed Portfolios

As it has just been demonstrated correct predictions do not in itself translate into agility. This means that agility must be handled explicitly, and to do this so-called Agility Factors are introduced. Agility Factors reflect the quality of units in the portfolio, so by introducing Agility Factors an ordering of units by quality can be achieved. Section 3.7 will then introduce different concepts for how to incorporate agility based on Agility Factors into algorithms for solving the Virtual Power Plant Dispatch Problem.

Notice that Agility Factors are defined in the way that is most intuitive for each type of unit. This means that for mixed portfolios sorting according to Agility Factors becomes slightly more involved, as summarized in Remark 1:

Remark 1*: (Sorting mixed portfolios according to Agility Factors) When mixed portfolios are sorted according to Agility Factors* Batteries *and* Bakeries *are sorted in increasing Agility Factor order followed by* Buckets *sorted in decreasing Agility Factor order. This way an ordering by quality is obtained.*

Though a bit unintuitive in practice the ordering specified in Remark 1 is easily implemented and does not in any way affect results or computation time.

The agility attributes of the *Bucket* are investigated in [1]. Here it is concluded that the Agility Factor of a *Bucket* should be the number of samples that it can operate at maximum power without becoming inactive/saturated. The Agility Factor of the *Bucket* is therefore defined as

Definition 4. The Agility Factor of Bucket i at sample k is

$$\mathcal{K}_i^{Bucket}(k) = \frac{\overline{E}_i - E_i(k)}{T_s \overline{P}_i}.$$

Batteries and *Bakeries* have quite different agility attributes than *Buckets*, because they have an energy requirement, which must be met. This means that as the deadline approaches *the Battery* or *the Bakery* will go from being a flexible resource to being a constraint. Consequently Agility Factors for *the Battery* and *Bakery* state how close we are (in terms of samples) to being forced to start the unit:

Definition 5. The Agility Factor of Battery i at sample k is

$$\mathcal{K}_i^{Battery}(k) = K_{End,i} - k - \frac{\overline{E}_i - E_i(k)}{T_s \overline{P}_i}.$$

Definition 6. The Agility Factors of Bakery i at sample k is

$$\mathcal{K}_i^{Bakery}(k) = K_{End,i} - K_{Run,i} - k.$$

3.4 Analytical Results

During the project several analytical contributions have been made relating to the computational complexity of the Virtual Power Plant Dispatch Problem. These contributions are summarized below.

The term *Information State* and *causality* were first introduced in [51]. Adapting to the notation of this Thesis these concepts are defined as follows:

Definition 7 (*Information State*)**.** The information state I_k at time k consists of

1. Parameters for all units $\{LU_i\}_{i=1,2,\ldots,N}$ at time k,
2. Energy level $E_i(k)$ for all units $\{LU_i\}_{i=1,2,\ldots,N}$ at time k, and
3. Realized values of $P_{Dispatch}$ up until time k, that is $P_{Dispatch}(0), P_{Dispatch}(1), \ldots, P_{Dispatch}(k)$.

Definition 8 (*Causality*)**.** A dispatch strategy is *causal* if its allocation at time k depends only on the information state at time k.

If *optimal, causal* dispatch strategies exist for a given dispatch problem, then this problem can be solved "one sample at a time", which significantly reduces the computational complexity. By Theorem 1 to 4, however, it is shown that *optimal, causal* dispatch strategies for *Buckets*, *Batteries and Bakeries* portfolios only exist in the very specific case where the portfolio consists of only *Buckets* all having $\underline{E} = \underline{P} = 0$.

Theorem 1. *Let $\{LU_i\}_{i=1,2,\ldots,N}$ denote a portfolio of Buckets all having $\underline{E} = \underline{P} = 0$. There* does *exist an optimal, causal dispatch strategy for $\{LU_i\}_{i=1,2,\ldots,N}$. This dispatch strategy can be obtained by solving a quadratic dispatch problem given by replacing the cost function in the Virtual Power Plant Dispatch Problem given by (3.16) to (3.17) with*

$$\max_{P_i(k)} \sum_{i=1}^{N} \frac{(\overline{E}_i - E_i(k) - P_i(k))^2}{-2\overline{P}_i}.$$

Proof. Omitted for brevity, see [2]. □

Theorem 2. *Let $\{LU_i\}_{i=1,2,\ldots,N}$ denote a portfolio Buckets. There does not exist an optimal, causal dispatch strategy for $\{LU_i\}_{i=1,2,\ldots,N}$.*

Proof. Proof is done by counterexample, [3]. Consider a portfolio consisting of the following two *Buckets*

$$\begin{aligned} \textit{Bucket}_1\colon\ & E_1(0) = 0, \\ & \overline{P}_1 = 1, \overline{E}_1 = 1, \\ & \underline{P}_1 = -1, \underline{E}_1 = -1, \end{aligned}$$

$$\begin{aligned} \textit{Bucket}_2\colon\ & E_2(0) = 0, \\ & \overline{P}_2 = 1, \overline{E}_2 = 3, \\ & \underline{P}_2 = -1, \underline{E}_2 = -3, \end{aligned}$$

Next define the following dispatch profiles

$$\begin{aligned} P^A_{Dispatch} &= (0, 2, 2), \\ P^B_{Dispatch} &= (0, -2, -2). \end{aligned}$$

Observe that it is possible to dispatch sequence $P^A_{Dispatch}$ in such a way that $\sum_{k=0}^{2} |S| = 0$. However, this is only achievable if $P_1(0) = -1$ and $P_2(0) = 1$. Observe also that equivalent arguments hold for $P^B_{Dispatch}$ if $P_1(0) = 1$ and $P_2(0) = -1$. At $k = 0$ a *causal* dispatch strategy must offer allocations of power based only on information available at time $k = 0$. Notice, however, that $P^A_{Dispatch}(0) = P^B_{Dispatch}(0)$ and since optimal dispatch of $P^A_{Dispatch}$ and $P^B_{Dispatch}$ requires different allocations at time $k = 0$, a *causal* dispatch strategy cannot exist. □

Theorem 3. *Let $\{LU_i\}_{i=1,2,\ldots,N}$ denote a portfolio of Batteries. There does not exist an optimal, causal dispatch strategy for $\{LU_i\}_{i=1,2,\ldots,N}$.*

Proof. See [51]. □

Theorem 4. *Let $\{LU_i\}_{i=1,2,\ldots,N}$ denote a portfolio of Bakeries. There does not exist an optimal, causal dispatch strategy for $\{LU_i\}_{i=1,2,\ldots,N}$.*

Proof. [3] Proof is done by counterexample. Consider a portfolio consisting of the following two *Bakeries*

$$\begin{aligned} \textit{Bakery}_1\text{: } & E_1(0) = 0, \\ & \overline{P}_1 = 1, \overline{E}_1 = 1, \\ & K_{run,1} = 1, K_{end,1} = 2, \\ \textit{Bakery}_2\text{: } & E_2(0) = 0, \\ & \overline{P}_2 = 3, \overline{E}_2 = 3, \\ & K_{run,2} = 1, K_{end,2} = 2. \end{aligned}$$

Next define the following dispatch profiles

$$\begin{aligned} P^A_{Dispatch} &= (2, 1), \\ P^B_{Dispatch} &= (2, 3). \end{aligned}$$

Observe that the optimal dispatch of either sequence $P^A_{Dispatch}$ or sequence $P^B_{Dispatch}$ to the portfolio has $\sum_{k=0}^{1} |S| = 1$. However, for $P^A_{Dispatch}$, this is only achievable if $P_1(0) = 0$ and $P_2(0) = 3$. For $P^B_{Dispatch}$ the required configuration is $P_1(0) = 1$ and $P_2(0) = 0$. The argumentation that a *causal* optimal dispatch strategy does not exist now follows as in the proof of Theorem 2. □

Since it has been showed that *optimal, causal* dispatch strategies do not generally exist, perfect prediction of $P_{Dispatch}$ is needed in order to determine the optimal dispatch at time k. Even with the assumption of perfect prediction of $P_{Dispatch}$ though, Theorem 5 below states that the dispatch problem is NP-complete if *Bakeries* are included in the portfolio. NP-completeness means that the computation time associated with finding an optimal solution using currently known algorithms grows extremely fast with the problem size.

Definition 9 (Subset-Sum Problem)**.** Let there be given a finite set $S \in \mathbb{N}$ and a target $T \in \mathbb{N}$. Is there a subset $S' \in S$ whose elements sum to T?

Lemma 1. *The Subset-Sum Problem is NP-complete.*

Proof. See [64]. □

Theorem 5. *Let $\{LU_i\}_{i=1,2,\ldots,N}$ denote a portfolio of Bakeries. Then problem (3.16) to (3.17) is NP-complete.*

Proof. Let $\boldsymbol{\mathcal{K}} \in \mathbb{N}_+$ and $K \in \mathbb{N}_+$ be given and assume without loss of generality that $\boldsymbol{\mathcal{K}} < K$. Next define portfolio $\{LU_i\}_{i=1,2,\ldots,N}$ consisting of all *Bakeries* having $\overline{P}_i = 1$, $K_{End,i} = K$ and $\sum_{i=1}^{N} K_{Run,i} = \boldsymbol{\mathcal{K}} + K$. Also define $P_{Dispatch}(k) = 2, k = 1, 2, \ldots, \boldsymbol{\mathcal{K}}$ and $P_{Dispatch}(k) = 1, k = \boldsymbol{\mathcal{K}} + 1, \boldsymbol{\mathcal{K}} + 2, \ldots, K$ (see Figure 3.15).

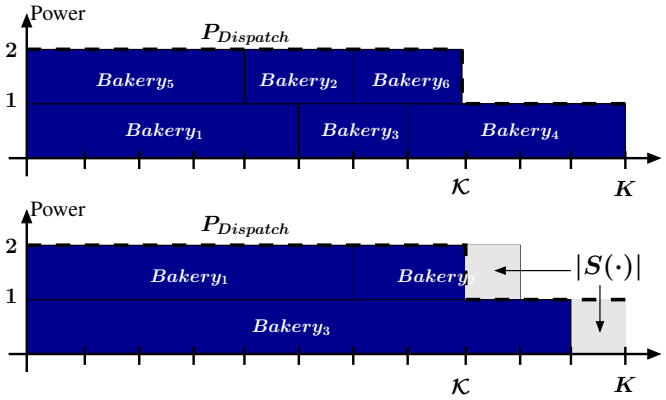

Figure 3.15: For the considered instance of the Virtual Power Plant Dispatch Problem a solution of problem (3.16) to (3.17) such that $f(P_i(\cdot)) = 0$ can exist if and only if there also exists a subset of $K_{Run,i}, i = 1, 2, \ldots, N$, which sums to $\mathcal{K}$.

To prove NP-completeness the following decision problem is formulated: Given the Virtual Power Plant Dispatch Problem instance constructed above does there exist a solution of problem (3.16) to (3.17) for which $f(P_i(\cdot)) = 0$?

First observe that $\sum_{k=1}^{K} P_{Dispatch}(k) = \mathcal{K}+K$ and $\sum_{i=1}^{N} K_{Run,i}\overline{P} = \sum_{i=1}^{N} K_{Run,i}$ $= \mathcal{K} + K$ as well. This means that exactly two *Bakeries* must be on at any sample until sample $\mathcal{K}$ and that exactly one *Bakeries* must be on at any sample after sample $\mathcal{K}$ in order for a solution with zero slack to exist. However, such a solution can exist if and only if there also exists a subset of $K_{Run,i}, i = 1, 2, \ldots, N$, which sums to $\mathcal{K}$.

This, however, corresponds exactly to the Subset-Sum Problem for the set $S = K_{Run,i}, i = 1, 2, \ldots, N$ and $T = \mathcal{K}$ since $\mathcal{K}$ and K are arbitrarily chosen positive integers. Thus, if there exists a polynomial time algorithm for solving the considered instance of the Virtual Power Plant Dispatch Problem then this algorithm could also solve the Subset-Sum problem in polynomial time. It now follows from Lemma 1 that the Virtual Power Plant Dispatch Problem is NP-complete for a portfolio of *Bakeries*. □

3.5 Exact Methods

Even though it has have proven that the computational complexity of the Virtual Power Plant Dispatch Problem is high, there are still options for achieving optimal solutions of problem (3.16) to (3.17). The question is whether these methods can be developed/tuned to solve large problem instances (>100 units) within a reasonable time frame. Achieving such solutions has been a significant, but not very successful part of the project. For the sake of completion, however, this section documents these efforts and specifically looks at solving the Virtual Power Plant Dispatch Problem by use of

- The commercial software package CPLEX,
- Dynamic Programming, and

- Dantzig Wolfe decomposition.

CPLEX [5]

In the industry a commonly used option for solving integer problems is to use the commercial software package CPLEX [65]. It has therefore been explored whether CPLEX can solve the Discrete Virtual Power Plant Dispatch Problem to optimality within a reasonable time frame. Computations are performed on a standard laptop.

The CPLEX performance is tested on two data sets: The first data set consists of ten randomly generated portfolios of 25 *Bakeries* and a simulation horizon of 100 samples. For this data set five of ten problems were solved successfully with an average computation time of 8 minutes. In the remaining five cases, however, computations are terminated with an error message stating that the computer has run out of memory and therefore no optimal solution is found.

The second data set consists of ten randomly generated portfolios of 50 *Bakeries* also with a simulation horizon of 100 samples. For this data set ten of ten problems terminated with the error message stating that the computer had run out of memory. Since all calculations finish due to lack of memory for 50 units and 100 samples, there is little hope that this option will scale to large problem instances.

Dynamic Programming

Dynamic Programming is an optimization technique which can be applied to problems, which exhibits optimal substructure and overlapping sub-problems. As will be demonstrated below the Virtual Power Plant Dispatch Problem does have these properties, so it has been investigated whether *Dynamic Programming* is an efficient method for solving the problem.

Optimal Substructure

Let Γ^* be an optimal solution of the Virtual Power Plant Dispatch Problem for a given portfolio of *Bakeries*. Then for each sample k the *Bakeries* can be partitioned into the sets of Finished, Running and Pending, see Figure 3.16. The task of scheduling the Finished *Bakeries* before sample k can then be considered a sub-problem of scheduling the entire portfolio before sample K. The solution specified by Γ^* for solving the sub-problem must, however, also be optimal if Γ^* is an optimal solution of the full problem. If there was a way of scheduling the Finished *Bakeries* with less slack before sample k than what is specified by Γ^*, then this sub-solution could be replaced in Γ^* to obtain a lower total cost. Then, however, Γ^* could not be optimal.

Algorithm

Each *Bakery* can be started at $K_{End} - K_{Run}$ different samples. This means that the total number of integer solutions is

$$\Pi_{i=1}^{N} K_{End,i} - K_{Run,i}. \tag{3.18}$$

A brute force algorithm would systematically generate all of these solutions and check, which one is optimal. At each sample, however, a *Bakery* can only have $K_{Run} + 2$

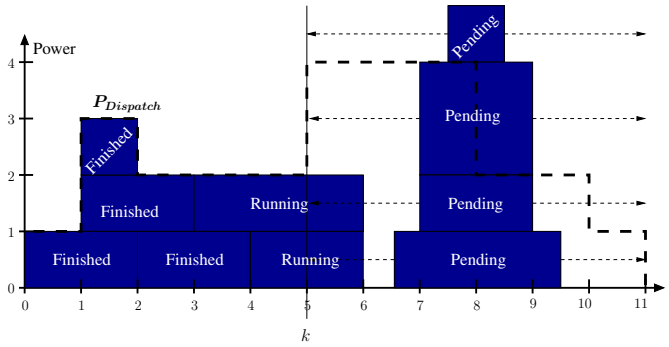

Figure 3.16: At sample k the *Bakeries* can be partitioned into the sets of Finished, Running and Pending.

different states, namely

1) Pending,
2) Started last sample,
3) Started two samples ago,
4) Started three samples ago,

$\vdots$

$K_{Run}+1$) Started K_{Run} samples ago,
$K_{Run}+2$) Finished,

so the number of states at sample k is only

$$\Pi_{i=1}^{N} K_{Run,i} + 2. \tag{3.19}$$

The number (3.19) still grows very fast with the size of the portfolio, but much slower than (3.18). This is exactly the insight, which is utilized in the *Dynamic Programming* algorithm described below.

Before sample 0 all *Bakeries* are in the Pending category. At sample 0 there are then 2^N combination for starting *Bakeries* and in the first stage of the algorithm all of these states are generated. This data set is denoted the state set. Now, it follows from the property of optimal substructure, that one of the generated combinations in the state set corresponds to the optimal way of starting *Bakeries* at sample 0, we just do not know yet, which one it is. The algorithm therefore generates a new state set for sample 1 based on the state set of sample 0.

An example is given in Figure 3.17 for a portfolio of two *Bakeries*. At sample zero, there a 2^2 combination for starting the *Bakeries*: Start both *Bakeries*, start $Bakery_1$ only, start $Bakery_2$ only or do not start any *Bakeries*. Each of these combinations correspond to a Pending, Started, Finished distribution for sample 1 and based on these the states for sample 1 can be generated as also depicted in Figure 3.17.

The size of the state set grows very quickly, but as time progresses there are more and more ways of arriving at the same state. This is also exemplified in Figure 3.17: At sample 2 three ways of putting both $\textit{Bakery}_1$ and $\textit{Bakery}_2$ in the Finished category have been found. Fortunately, however, it is only necessary to store the best way of arriving at this state and in our case this corresponds to starting $\textit{Bakery}_1$ at sample 1 and $\textit{Bakery}_2$ at sample 0. Due to the property of optimal substructure the other two options cannot be optimal and are therefore discarded.

For larger portfolios and longer simulation horizons the size of the state set therefore stops growing at some point, see Figure 3.18, because more and more overlapping sub-problems are found and only the best combination has to be stored for further investigation.

Minimizing the State Set

To speed up computations the state sets depicted in Figure 3.18 should be as small as possible. To cut down the set two bounds are derived:

- For a given state and a given sample k a lower bound on the cost of scheduling the Pending *Bakeries* after sample k, and
- An upper bound on the optimal solution.

If the lower bound on the total cost associated with a given state is larger than an upper bound of the optimal solution, then this state can obviously not lead to an optimal solution. It can therefore be discarded, which reduces the remaining computations.

A lower bound on the cost of scheduling the Pending *Bakeries* of given state can be obtained as follows: At sample k assume that a portfolio is given with a Finished, Running, Pending distribution. A lower bound, $\Phi(k)$, on the cost of scheduling the pending *Bakeries* after sample k is then given by

$$\Phi(k) = \left| \sum_{k}^{K} \left(\left| P_{Dispatch}(k) - \sum_{Running\ at\ k} \overline{P}(k) \right| \right) - \sum_{Pending} \overline{E} \right|.$$

This corresponds to ignoring the run time and constant power constraints of the *Bakeries*. In the best case the energy remaining in $P_{Dispatch}$ after sample k and the energy required by the pending *Bakeries* will be able to map together perfectly except for differences in absolute magnitude between two.

An upper bound on the optimal solution is obtained by running a heuristic algorithm parallel to the *Dynamic Programming* algorithms. In the implementation the heuristic algorithm *GRASP Sorted* presented in Section 3.7 is started in a parallel thread. This means that a tighter and tighter upper bound on the optimal solution is generate over time.

Based on the lower bound of a given state and the upper bound of the optimal solution states are checked for possible optimality before they are added to the state set.

Results

The following simulation example considers randomly generated problem instances with a simulation horizon $K = 10$ and portfolios of ten to fifteen *Bakeries*. In all simulations

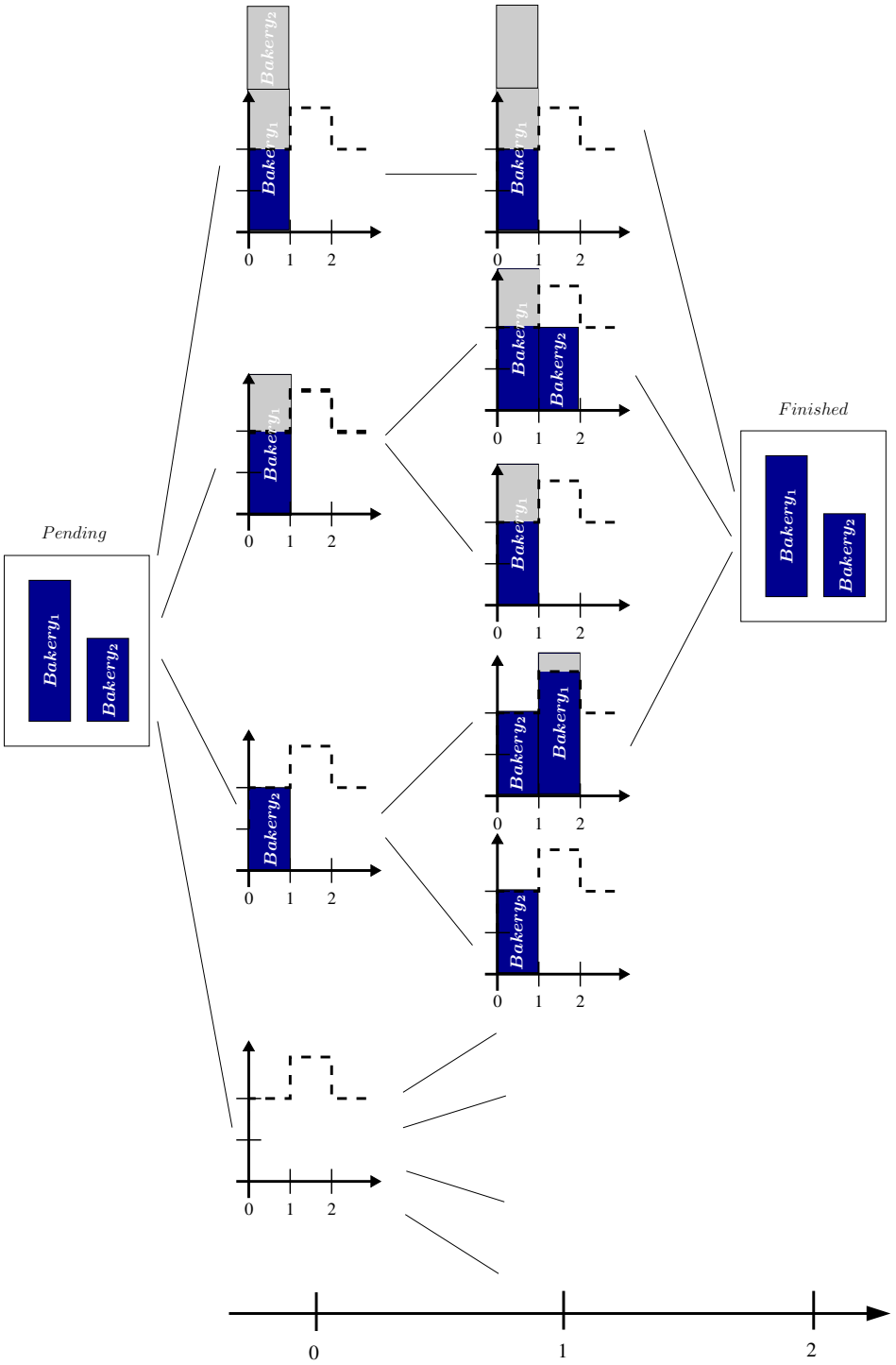

Figure 3.17: Example of state sets generated by *Dynamic Programming* algorithm.

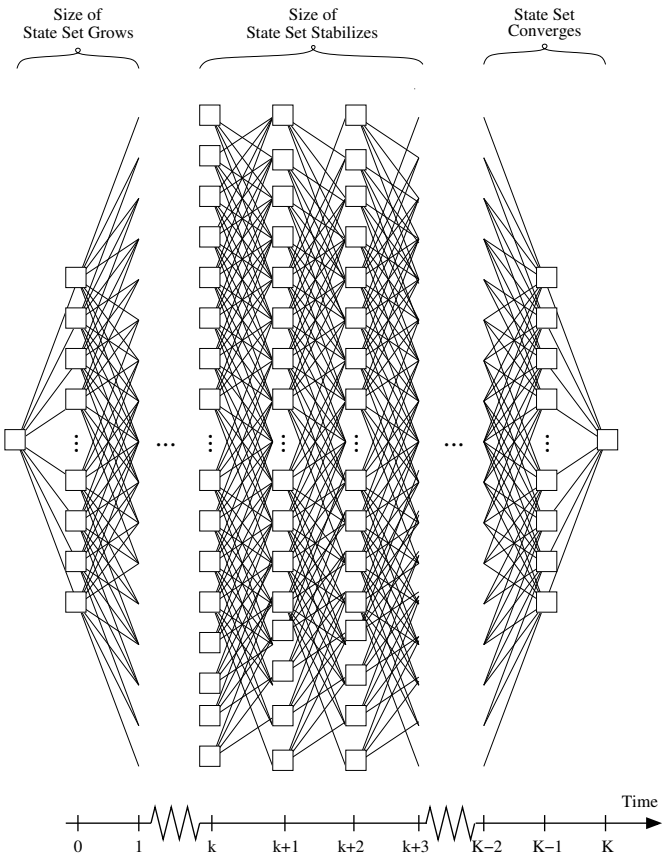

Figure 3.18: Size of state sets generated by *Dynamic Programming* algorithm.

$T_s = 1$ and $E_{i,0} = 0$ for all units. The *Dynamic Programming* algorithm has been implemented in C#, [66]. Computations are performed on a standard laptop PC.

In Figure 3.19 the average computation time for the *Dynamic Programming* algorithm is depicted as a function of portfolio size for a simulation horizon fixed at $K = 10$. It can be seen that the computation time grows faster than exponentially and for just 15 units the average computation time is nearly five minutes. Again there is therefore little hope that this option will scale to large problem instances.

Dantzig-Wolfe Decomposition

This section investigates the usefulness of the *Dantzig-Wolfe* decomposition for solving the Virtual Power Plant Dispatch Problem. *Dantzig-Wolfe* is a decomposition method, which can be used to solve linear programs. It is especially useful for problems with block-angular structure and as will be illustrated later the Virtual Power Plant Dispatch Problem does have block-angular structure. Furthermore, *Dantzig-Wolfe* decomposition can in some cases be used to solve linear mixed integer problems e.g. the cutting stock problem (see [67]). It was therefore investigated early in the project whether *Dantzig-*

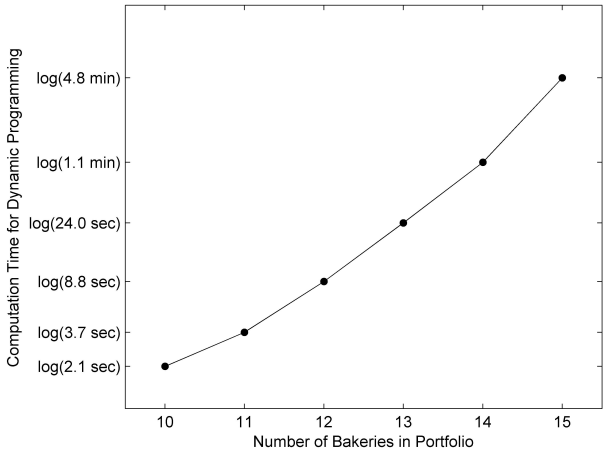

Figure 3.19: Computation time as a function of problem size for the *Dynamic Programming* implementation.

Wolfe decomposition was suitable for solving the Virtual Power Plant Dispatch Problem. It was found that *Dantzig-Wolfe* decomposition is indeed useful for solving the Virtual Power Plant Dispatch Problem if only *Buckets* and *Batteries* are included, but that the method cannot handle *Bakeries*.

The theoretical explanation why *Dantzig-Wolfe* decomposition can be used to solve the Virtual Power Plant Dispatch Problem when only *Buckets* and *Batteries* are considered is that in this case the problem is a nice, benign linear program. When *Bakeries* are added, however, the problem becomes a much more malignant mixed integer program, and although *Dantzig-Wolfe* decomposition is suitable for solving some of these, the method is unfortunately not well suited for solving the Virtual Power Plant Dispatch Problem.

This Section will document the results described above. Rather than a strict theoretical description and proof of convergence of the *Dantzig-Wolfe* decomposition method (see [19] and [68] for those results) the structure of the method will be described more intuitively. To further simplify the explanation agility will not be included in the problem formulation. When explaining the *Dantzig-Wolfe* decomposition method a portfolio of *Batteries* is first used to illustrate how *Dantzig-Wolfe* decomposition will generate a solution for the Virtual Power Plant Dispatch Problem. After that it is illustrated how and why the method will stall when *Bakeries* are added to the portfolio.

The *Dantzig-Wolfe* method relies on the following three key concepts

1. Iterative solution method for problems with block-angular structure,
2. Shadow prices and column generation, and
3. Convex combinations.

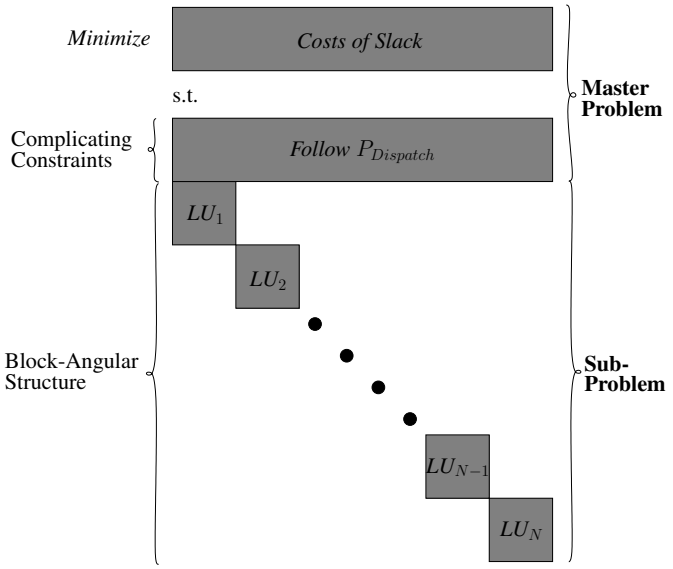

Figure 3.20: Block-angular matrix structure of the Virtual Power Plant Dispatch Problem.

Iterative Solution Method for Problems with Block-Angular Structure

A problem with block-angular structure is a problem where a number of independent systems must cooperate to meet a common goal or share a scarce resource. The block-angular structure of the Virtual Power Plant Dispatch Problem is depicted in Figure 3.20. Here each block in the diagonal of the constraint matrix describes the constraints and dynamics of a local unit. The common goal of following $P_{Dispatch}$ is given by the transverse constraints. Since $P_{Dispatch}(k)$ is considered for $k = 1, 2, \cdots, K$ in Problem (3.16) to (3.17) there are K complicating constraints, one for each considered time step. The transverse constraints are also denoted the complicating constraints. This name is appropriate, because if the cross-portfolio requirements were not present the problem could simply be completely separated into the N much smaller block-problems. These could then be solved in parallel on several CPUs to really speed up computations.

The strategy applied by the *Dantzig-Wolfe* decomposition methods is to split the full problem into a Master Problem and a Sub-Problem. The Master Problem consists of the cost function and the complicating constraints and the Sub-Problem is the block-angular constraints, see Figure 3.20. Finding an optimal solution of the full problem is then accomplished through an iterative procedure where a Master Problem and a Sub-Problem is solved once per iteration, see Figure 3.21. The *Dantzig-Wolfe* decomposition method is especially useful for problems with block-angular structure, because the Sub-Problem can apply the strategy discussed earlier of solving all blocks in parallel since the complicating constraints have been eliminated in the sub-problem. The *Dantzig-Wolfe* decomposition method is, however, not only applicable for problems with block-angular structure.

The Master Problem and the Sub-Problem are both updated at each iteration and it can be proven that the solutions of the Master Problem converges to the solution of full problem, see [19] and [68].

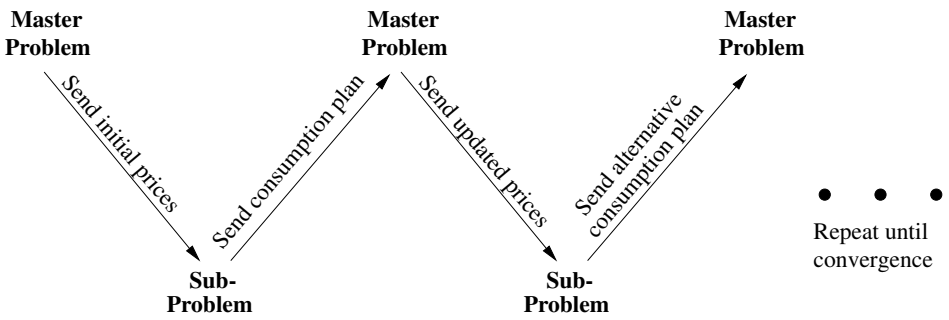

Figure 3.21: The *Dantzig-Wolfe* decomposition method solves the full optimization problem through an iterative procedure where a Master Problem and a Sub-Problem is solved once per iteration. Iterations continue until the solution of the Master Problem has converged to the optimal solution of the full problem.

Shadow Prices and Column Generation

As depicted in Figure 3.21 there is an interface between the Master Problem and Sub-Problem in which the Master Problem sends prices to the Sub-Problem and the Sub-Problem returns a consumption plan to the Master Problem. For our problem this interface summarized shadow prices and column generation.

The prices, which the Master Problem sends to the Sub-Problem, are the so-called shadow prices or Lagrange multipliers for the complicating constraints[1]. The shadow prices of a constraint measures the rate at which the objective would change if the constraint was slightly relaxed, see [64]. This means that the shadow price is a sort of shared measure of the cost function and complicating constraints. In our case, since there are K complicating constraints (one for each time step) there are also K shadow prices.

Based on the prices received from the Master Problem the sub-problem generates a new consumption plan. A high price associated with a time step is therefore a signal from the Master Problem to the Sub-Problem that the Sub-Problem should generate a consumption plan, with a high value at this sample. The consumption plans generated by the Sub-Problem are always feasible, but not necessarily optimal. The new consumption plans are communicated back to the Master Problem and here they are denoted columns. The name column generation therefore stems from the fact that a new column is generated once per iteration.

Convex Combinations of *Batteries*

To understand convex combinations first consider a portfolio of four *Batteries* with parameter values $\overline{P} = 2$ and $\overline{E} = 2$ and $P_{Dispatch}$ as depicted in Figure 3.22.

As just explained, the Master Problem receives a new consumption plan from the Sub-Problem once per iteration. At each iteration the Master Problem then computes a solution of the full problem as a convex combination of the consumption plans generated so far. An example of a convex combination of consumption plans is given in Figure

[1] Since the objective is to determine the shadow prices of the complicating constraints it is really the dual Master Problem rather than the primal, which is solved.

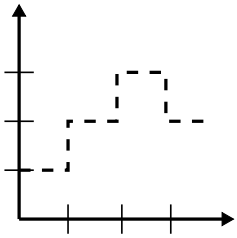

Figure 3.22: $P_{Dispatch}$ considered to explain convex combinations.

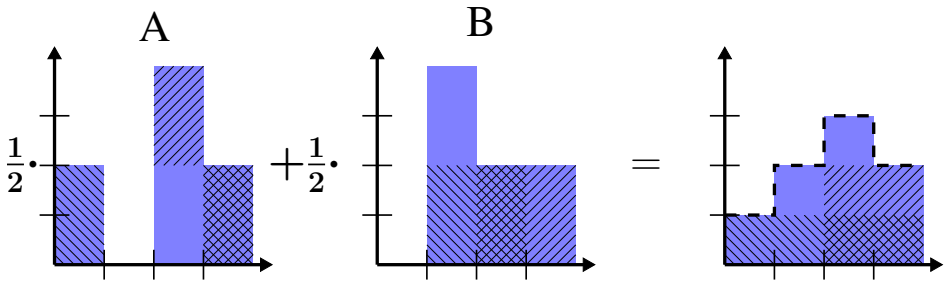

Figure 3.23: A convex combination of feasible dispatch plans of *Batteries* is also a feasible dispatch plan.

3.23. The considered portfolio consists of four *Batteries* all having $\overline{P} = 2$ and $\overline{E} = 2$. For this portfolio A and B in Figure 3.23 are both feasible dispatch plans, which could have been generated by the Sub-Problem. A convex combination of A and B is then a linear combination of A and B for which the fraction of A plus the fraction of B is less than or equal to one; that is $\alpha_A A + \alpha_B B$ is a convex combination of A and B if $\alpha_A + \alpha_B \leq 1$.

It can be proven that when the solution space is continuous, closed and bounded then any convex combination of feasible columns is also a feasible column (see [19]). When considering *Batteries* (or *Buckets*), these requirements are satisfied. Therefore consider the portfolio of four *Batteries* with $\overline{P} = 2$ and $\overline{E} = 2$ and assume that A and B have been generated by the Sub-Problem. Since the Master Problem is attempting to solve the full problem, that is balance $P_{Dispatch}$ in Figure 3.22 as closely as possible, the Master Problem will generate the convex combination of A and B given on the far right of Figure 3.23. Here the slack is zero, so an optimal solution has been determined and the iteration procedure will terminate.

Convex Combinations of *Bakeries*

We have now obtained the necessary understanding of the *Dantzig-Wolfe* decomposition method to see why the method is not suitable for solving the Virtual Power Plant Dispatch Problem for *Bakeries*. Lets consider a portfolio of four *Bakeries* also having $\overline{P} = 2$ and $\overline{E} = 2$. This means that the *Bakeries* must have a runtime of one sample at power consumption $P = 2$. consequently dispatch plans A and B in Figure 3.24 could be

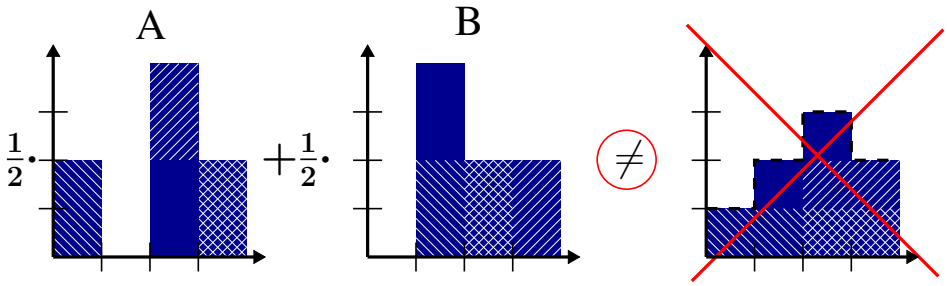

Figure 3.24: A convex combination of feasible dispatch plans for *Bakeries* is *not* necessarily a feasible dispatch plan. In this case the *Bakeries* must have a runtime of one sample at power consumption $P = 2$. Consequently A and B are feasible dispatch plans, but the convex combination does not respect the integer constraints and is therefore infeasible.

generated by the Sub-Problem even when the Sub-Problem respects the integer constraint (3.15).

The problem arises because the Master Problem will generate the convex combination of A and B given on the far right of Figure 3.24 and terminate since it seems that an optimal solution of the full problem has been determined. The problem is, however, that the convex combination in Figure 3.24 is not feasible since it does not respect the minimum runtime constraint (3.15). This illustrate why the *Dantzig-Wolfe* decomposition method is not suitable for solving the Virtual Power Plant Dispatch Problem for *Bakeries*.

3.6 Agility in Dispatch 2

In Section 3.3 it was explained how to add Agility Factors to a portfolio and obtain an ordering by quality. This section returns to the subject of agility in dispatch and explains how to incorporate agility directly into the Virtual Power Plant Dispatch Problem formulation given by equation (3.16) to (3.17). To do this let $\{LU_i\}_{i=1,2,\ldots,N}^{Sorted}$ denote a set of local units, *which is sorted according to Agility Factors as specified by Remark 1.* Then associate an agility weight, $w_{i,k} \in \mathbb{R}$, with each unit and time step and incorporate these into the cost function (3.16) as follows:

$$\min_{P_i(\cdot)} \sum_{k=1}^{K} \left(|\mathcal{S}(k)| + \sum_{i=1}^{N} w_{i,k} |P_i(k)| \right). \tag{3.20}$$

The agility weights should now be chosen such that local unit i is dispatched before local unit j, if local unit i is less agile than local unit j. To achieve this agility weights should add a penalty to increasing the energy term of each local unit. This penalty should be proportional to the unit index after sorting, so the cost function (3.20) is replaced with

$$\min_{P_i(\cdot)} \sum_{k=1}^{K} \left(|\mathcal{S}(k)| + \sum_{i=1}^{N} i |E_i(k)| \right). \tag{3.21}$$

However, since $E_i(k) = \sum_{l=1}^{k} T_s P_i(l)$ Equation (3.21) can be written as

$$\min_{P_i(\cdot)} \sum_{k=1}^{K} \left(|\mathcal{S}(k)| + \sum_{i=1}^{N} i \ (K+1-k) \ T_s \ |P_i(k)| \right).$$

The agility weights are therefore given by

$$w_{i,k} = i \ (K+1-k) \ T_s \tag{3.22}$$

where i is the unit index after sorting according to Remark 1.

By adding agility to the Virtual Power Plant Dispatch Problem, the solutions found will be more robust of prediction errors and short prediction horizons as explained in Section 3.3. A further modification of the problem formulation is obtained by adding so-called impatience weights to the cost of slack, as

$$\min_{P_i(\cdot)} \sum_{k=1}^{K} \left(w_k |\mathcal{S}(k)| + \sum_{i=1}^{N} w_{i,k} |P_i(k)| \right). \tag{3.23}$$

$$s.t.$$

$$\sum_{i=1}^{N} P_i(k) + \mathcal{S}(k) = P_{Dispatch}(k). \tag{3.24}$$

where $w_{k_1} > w_{k_2}$ if $k_1 < k_2$. Adding the impatience weights, w_k, to the cost function ensures that if the problem cannot be solved without introducing slack, then imbalances will appear as late on the prediction horizon as possible.

In the remaining sections the formulation (3.23) to (3.24) of the Virtual Power Plant Dispatch Problem will be used and it will be denoted the Virtual Power Plant Dispatch Problem with Agility.

3.7 Heuristic Methods

It has now been demonstrated that several efforts to determine exact optimal solutions of larger instances of the Virtual Power Plant Dispatch Problem have not been fruitful. It has also been proven in Theorem 1 that the Virtual Power Plant Dispatch Problem is NP-complete. In this section we will therefore explore the alternative to exact optimization methods, which is to use heuristic optimization in order to determine solutions, which are sub-optimal, but fast to compute.

Intuitively the idea of heuristic optimization seems applicable: As seen in the case of dynamic programming the computation time grows faster than exponentially with the problem size. So lets assume that the algorithm could be tuned and run on a fast enough computer to make the it useful in real time for, say, 1.000.000 consumption units. If at this point, however, just one more unit was added to the portfolio, then all efforts would be in vain as computation time would again explode. From a more application-oriented point of view, however, the consumption of the $1.000.001^{th}$ unit is irrelevant compared to the remaining portfolio. The marginal computation time is therefore growing exponentially with the number of units while the marginal significance is vanishing.

In this section the option of solving the Virtual Power Plant Dispatch Problem with Agility using heuristic optimization will therefore be investigated. In the first efforts the focus is on *Bakeries* and it is explained how the two methods known in literature as *Hill Climber* , [69], and *GRASP* (Greedy, Randomized, Adaptive Search Procedure), [70], can be adapted to solve the Virtual Power Plant Dispatch Problem with Agility for a portfolio of *Bakeries*.

Next focus is on developing fast algorithms for solving the Virtual Power Plant Dispatch Problem with Agility for mixed portfolios. This is done by exploring the relationship between prediction and agility further and by developing sorting-based algorithms for solving the problem.

Hill Climber and GRASP

In this section the performance of *Hill Climber* and *GRASP* for solving the Virtual Power Plant Dispatch Problem with Agility will be investigated. Other heuristic methods that could also be considered for solving the problem are Ant Swarm Optimization [58], Genetic Algorithms [69] and Tabu Search [71]. These algorithms are, however, all based on manipulating a set of candidate or tabu solutions. They thus have a tendency to drown in logistics as the majority of the available computation time is spent running through and updating solution sets.

As explained above heuristic optimization methods are used to determine solutions, which are sub-optimal, but fast to compute. For *Hill Climber* and *GRASP* this means that the longer the method runs the better quality solution it will find. Computation time is therefore a parameter and the methods will terminate when the time limit is reached and return the best solution encountered so far.

The generic descriptions of the *Hill Climber* and *GRASP* algorithms goes as follows:

- *Hill Climber*, [69]: To solve an optimization problem the *Hill Climber* method first generates a random initial solution for the considered problem. Next it is attempted to improve the current solution through some update procedure. If the cost of the updated solution is less than the cost of the current solution, then the updated solution will take place as current solution. This procedure is continued until the time limit is reached.

- *GRASP*, [70]: To solve an optimization problem the *GRASP* method constructs an initial solution one element at a time by use of a greedy algorithm. The choice of next element to be added to the initial solution is determined by constructing a candidate list of most beneficial choices. The probabilistic element in *GRASP* is then introduced by randomly choosing one of the candidates in the candidate list, but not necessarily the top candidate. After an initial solution has been generated *Hill Climber* is called to achieve a further improvement of the solution.

In the remainder of this section the generic algorithms of *Hill Climber* and *GRASP* are tailored to the Virtual Power Plant Dispatch Problem. This results in four new methods and the choice of initial solution and update procedure for these methods are summarized in Table 3.5. Pseudo-code for the algorithms are given in [5]. All algorithms are explained below, but as they are all based on the idea of an n-move that concept is explained first.

Method	Initial Solution	Update Procedure
Uniform Selection Hill Climber	Uniform Probability	Uniform Probability
Weighted Selection Hill Climber	Uniform Probability	Weighted Probability
GRASP Random	Weighted Probability	*Hill Climber*
GRASP Sorted	Agility Sorted	*Hill Climber*

Table 3.5: Choice of initial solution and update procedure for the tailored versions of *Hill Climber* and *GRASP*.

n-move

The idea of an n-move is central to the *Hill Climber* and *GRASP* adaptations described below. When considering only *Bakeries* a solution of the Virtual Power Plant Dispatch Problem with Agility is described by a set of feasible start samples, one for each *Bakery* in the portfolio. An n-move consists of changing the start sample of any n units to other feasible locations. An example of a 3-move is depicted in Figure 3.25. Here the start sample of $Bakery_1$ is changed from 5 to 6, the start sample of $Bakery_3$ is changed from sample 1 to 4 and the start sample of $Bakery_4$ is changed from sample 0 to 5.

Uniform Selection Hill Climber and Weighted Selection Hill Climber

To adapt the generic *Hill Climber* method to solve the Virtual Power Plant Dispatch Problem with Agility it must be specified how to generate the initial solution and what sort of update procedure to use.

The first version of the *Hill Climber* method is denoted *Uniform Selection Hill Climber*. *Uniform Selection Hill Climber* generates an initial solution by randomly choosing feasible start samples for each *Bakery*. This is a very fast procedure for generating an initial solution, but it also means that the initial solution has no similarity to $P_{Dispatch}$. We will return to this problem later.

In *Uniform Selection Hill Climber* the update procedure consist of a random n-move. The n-move is performed by choosing n *Bakeries* from the portfolio with uniform probability and then choosing feasible start samples for each *Bakery* also with uniform probability.

The second version of the *Hill Climber* method is denoted *Weighted Selection Hill Climber*. This version also generates the initial solution by randomly choosing feasible start samples for each *Bakery*. When performing the n-move in the update procedure, however, start samples are given higher priority if they give a relatively larger improvement of the objective.

The difference between the two methods is depicted in Figure 3.26 and Table 3.6: Suppose that at some iteration the distribution in the top illustration on Figure 3.26 has been obtained. Suppose also that a 1-move has to be performed of $Bakery_4$ and that $Bakery_4$ has deadline at sample 9. For *Uniform Selection Hill Climber* samples 0 to 8 will be chosen as new start samples with equal probability of $\frac{1}{9}$ since there are nine remaining options for starting $Bakery_4$. For *Weighted Selection Hill Climber* the calculations are

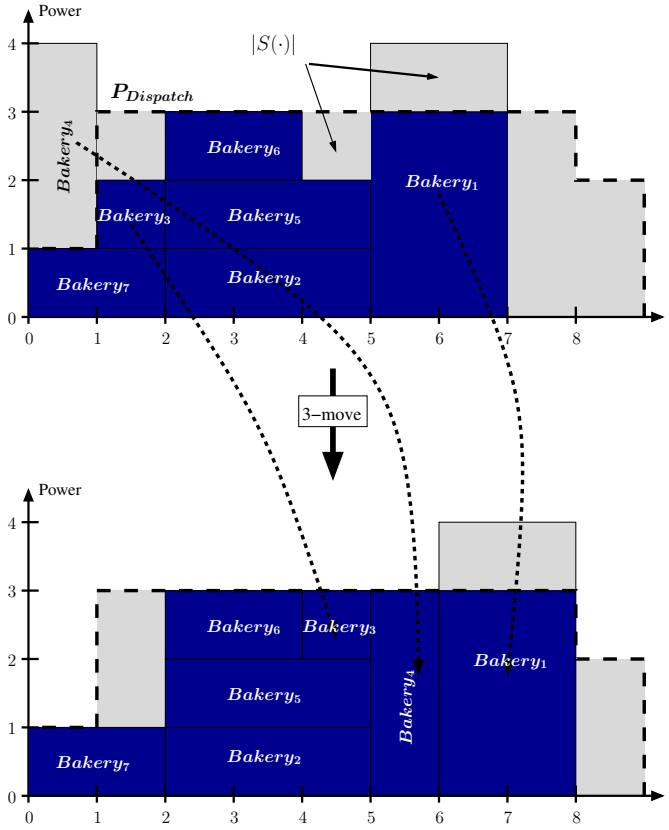

Figure 3.25: Example of a 3-move.

slightly more involved: If agility and impatience weights are ignored[2] then the slack of the solution depicted in the top illustration on Figure 3.26 is 12. The slack will drop to 10 if $Bakery_4$ is started at sample 1 or 4 and drop to 6 and 8 respectively if $Bakery_4$ is started at sample 7 or 8. Given that a 1-move has to be performed of $Bakery_i$ *Weighted Selection Hill Climber* thus chooses sample k with probability relative to the benefit of starting at each feasible sample, that is

$$p_{WeightedSelection,i}(k) = \frac{\Delta Cost(k)}{\sum_{m=0}^{K_{End,i}-K_{Run,i}} \Delta Cost(m)}.$$

In the example illustrated by Figure 3.26 and Table 3.6 the sample probabilities for performing a 1-move of $Bakery_4$ will thus be $\frac{2}{14}$ for sample one or four and $\frac{6}{14}$ and $\frac{4}{14}$ respectively for sample 7 and 8. Consequently it is much more likely that *Weighted Selection Hill Climber* will choose the best available option of sample 7 for starting $Bakery_4$ than it is that *Uniform Selection Hill Climber* will make the same choice.

It might seem from the description above that *Weighted Selection Hill Climber* will always perform better than *Uniform Selection Hill Climber* since it makes more intel-

[2] Agility and impatience weights are included in the simulation results presented below.

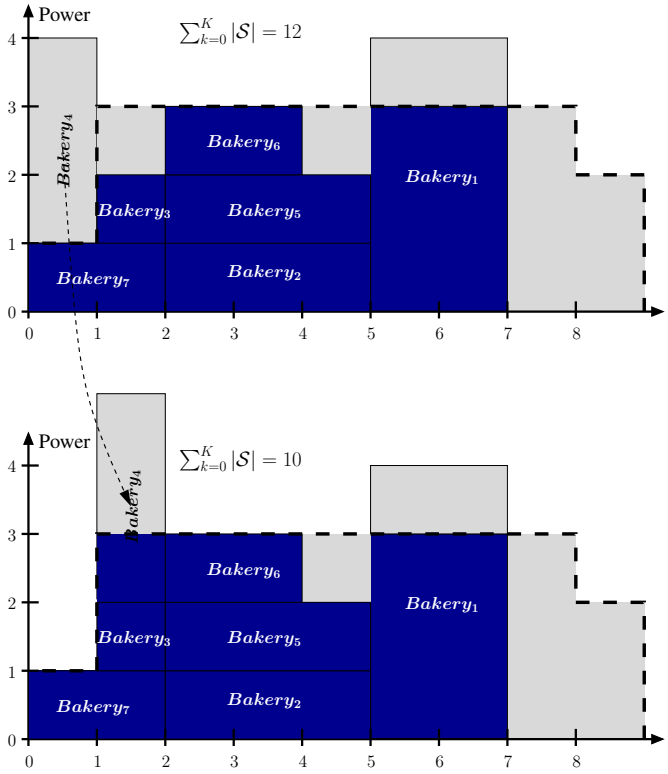

Figure 3.26: Example used to illustrate probability of *Uniform Selection Hill Climber* and *Weighted Selection Hill Climber* respectively choosing each sample as new start time when performing a 1-move of $Bakery_4$. Probabilities are given in Table 3.6.

Sample	0	1	2	3	4	5	6	7	8	Sum
ΔCost	0	-2	0	0	-2	0	0	-6	-4	-14
$p_{UniformSelection}$	$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$	$\frac{1}{9}$	1
$p_{WeightedSelection}$	0	$\frac{1}{7}$	0	0	$\frac{1}{7}$	0	0	$\frac{3}{7}$	$\frac{2}{7}$	1

Table 3.6: Probability of *Uniform Selection Hill Climber* and *Weighted Selection Hill Climber* respectively choosing each sample as new start time when performing a 1-move of $Bakery_4$ in Figure 3.26.

ligent choices about when to start *Bakeries*. The drawback of *Weighted Selection Hill Climber* is, however, that it has to spend time computing differentiated probabilities. This means that fewer iterations can be performed within the time limit. To determine whether computation time is best spend on performing more random n-moves or on computing differentiation probabilities the two methods must be implemented, tuned and tested. This is done in the Results section below.

GRASP Random and GRASP Sorted

A weakness of the *Hill Climber* methods described above is that the initial solution is generated by choosing both *Bakeries* and start samples with uniform probability. This means that the initial solution has absolutely no similarity to $P_{Dispatch}$. This problem is mended by *GRASP*.

Again two adapted versions of the algorithm have been developed, namely *GRASP Random* and *GRASP Sorted*. *GRASP Random* falls closest to the generic description of the *GRASP* algorithm, but as illustrated later it has some challenges related to the Virtual Power Plant Dispatch Problem with Agility. To address this issue *GRASP Sorted* is developed.

To adapt the generic *GRASP* method to solve the Virtual Power Plant Dispatch Problem with Agility it must be specified how to construct the candidate list and how to choose an element from the list. In *GRASP Random* this is done based on two lists: The Unit list and the Candidate List. The Unit List consist of m *Bakeries*, which are chosen randomly from the portfolio with uniform probability. For each *Bakery* in the Unit List the relative benefit of starting at each feasible sample is again computed as exemplified in Figure 3.26 and Table 3.6. The Candidate List is then constructed as the l most beneficial *Bakery*-sample combinations. This means that the Candidate list consist of the l best sample choices for starting the randomly chosen *Bakeries* in the Unit List. Finally, the next element to add to the initial solution is chosen from the Candidate List with uniform probability. After the initial solution has been generated *Uniform Selection Hill Climber* or *Weighted Selection Hill Climber* is called to achieve a further improvement of the solution.

When exploring *GRASP Random* it was discovered that the method generates initial solutions, which overshoot $P_{Dispatch}$ in the beginning of the simulation horizon and fall lower than $P_{Dispatch}$ towards the end of the horizon. This behaviour can be explained as follows:

When the impatience weights defined in Section 3.6 are added to the problem it means that slack becomes ”cheaper” towards the end of the horizon than in the beginning. When *GRASP Random* builds the initial solution one *Bakery* at a time it will therefore initially start units early on the simulation horizon. Now, at some point a good match with $P_{Dispatch}$ is obtained for, say, the first 10 samples. However, if a *Bakery* then remains in the portfolio at this point, which has deadline 10 or less, then it can only be added to the initial solution in such a way that it makes the accumulated power consumption overshoot $P_{Dispatch}$ somewhere in the first 10 samples. When this has happened a number of times (see Figure 3.27) the result is a consumption profile, which overshoots $P_{Dispatch}$ in the beginning of the horizon and falls lower than $P_{Dispatch}$ towards the end of the horizon.

GRASP Sorted is developed to alleviate this problem. The algorithm is identical to *GRASP Random* except that when generating the Unit List, *Bakeries* are not chosen with uniform probability. Instead units are sorted according to Agility Factors as specified in Remark 1. The Unit List then consists of the m least agile *Bakeries*. The Candidate List is again generated based on the Unit List as with *GRASP Random* and also the next element to add to the initial solution is chosen from the Candidate List with uniform probability. Again, after the initial solution has been generated *Uniform Selection Hill Climber* or *Weighted Selection Hill Climber* is called to achieve a further improvement of the solution.

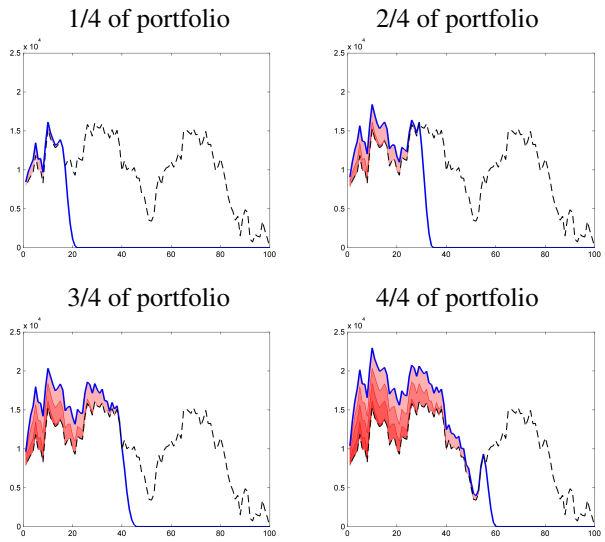

Figure 3.27: *GRASP Random* builds an initial solution one *Bakery* at a time and above it is depicted how the initial solution looks, when 1/4, 2/4, 3/4 and 4/4 of the *Bakeries* in a portfolio have been added.

Since slack is cheaper towards the end of the horizon, *GRASP Random* will first start units early in the horizon. When 1/4 of the portfolio has been added a decent fit with $P_{Dispatch}$ has been obtained for sample 0 to 18. However, there are still *Bakeries* remaining in the portfolio, which have deadline 18 or less, and now *GRASP Random* can only add these in such a way that the accumulated power consumption before sample 18 overshoots $P_{Dispatch}$ even further. This means that as *Bakeries* are added more and more positive slack builds up at the beginning of the simulation horizon, as can be seen from the progression of the figures.

Results

Before the algorithms can be compared they must be properly tuned. Tuning results can be found in [5]. The algorithms have been implemented in C# [66], computation time is fixed at 10 seconds and calculation are performed on a standard laptop PC. The algorithms are tested in randomly generated portfolios of 1.000, 10.000 and 100.000 *Bakeries* and a simulation horizon of 100 samples. In order to compare performance on problem instances with very different absolute values the percentage gap and percentage deviation are calculated. Since no procedure has been found to determine the actual optimal solutions of problem instances of the considered size, the percentage gap and the percentage deviation is computed relative to the minimum value found over all calculations on each particular problem instance.

It is found that for all problem sizes the *Hill Climber* methods and *GRASP Random* have very similar performance, with no clear winner. *GRASP Sorted*, on the other hand, outperforms all the other methods by at least an order of magnitude both in terms of percentage gap and percentage deviation and for all problem sizes.

	1.000 *Bakeries*		10.000 *Bakeries*		100.000 *Bakeries*	
	E	σ	E	σ	E	σ
Uniform Select.	32.4	7.7	58.6	3.6	47.8	1.5
Weighted Select.	38.0	8.6	62.5	4.5	59.3	1.7
GRASP Random	23.7	3.1	65.3	2.5	32.9	1.4
GRASP Sorted	0.6	0.4	2.5	0.3	0.7	0.5

Table 3.7: Percentage gap and percentage deviation for *Uniform Selection Hill Climber*, *Weighted Selection Hill Climber*, *GRASP Random* and *GRASP Sorted*.

Method	Type	First Presented	Based on
Predictive-Balancing	Pure	[3]	Moving Horizon
Agile-Balancing	Pure	[3]	Sorting
Predictive-Balancing-with-Agility	Hybrid	Novel	Moving Horizon
Agile-Balancing-with-Prediction	Hybrid	Novel	Sorting

Table 3.8: Overview of methods.

Figure 3.28 depicts solutions found using each algorithm for a randomly generated simulation problem with 100.000 *Bakeries*. The agility of the solutions is illustrated by green, red, cyan and magenta lines, which is the accumulated consumption of the first, second, third and fourth quarter of *Bakeries*, respectively, when the portfolio is sorted according to agility as specified in Remark 1. A visual inspection confirms the general conclusions that *Uniform Selection Hill Climber*, *Weighted Selection Hill Climber* and *GRASP Random* have very similar performance, as the curves can hardly be distinguished. However, *GRASP Sorted* is clearly superior. It can be seen that particularly in the beginning of the simulations *GRASP Sorted* has less slack than the other methods and *GRASP Sorted* has furthermore found a far more agile solution, which can be seen by the very steep slopes of the quarter lines for this method.

Prediction & Agility

In this section the objective is to develop heuristic optimization methods specifically designed to have short computation times. We will do this by investigating the importance of prediction and agility further and present different ways of combining the two.

This section will present four heuristic methods (see Table 3.8) for solving the Virtual

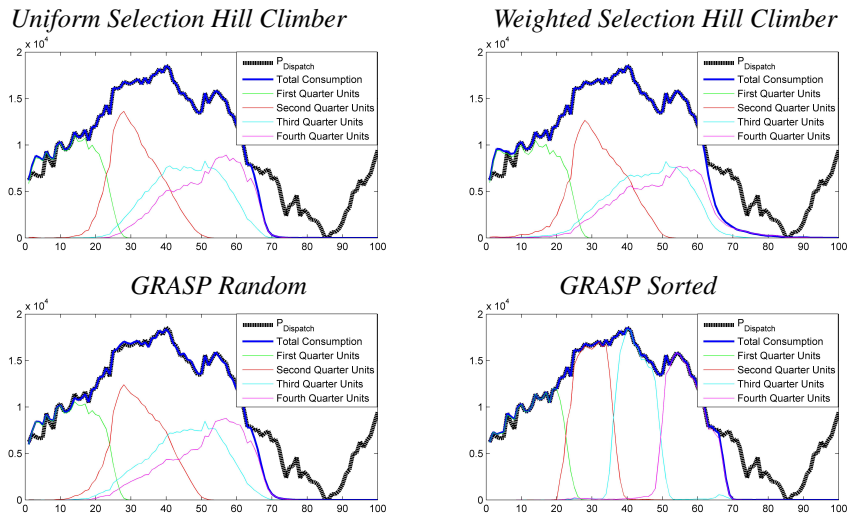

Figure 3.28: Solution of problem instance of the Virtual Power Plant Dispatch Problem with 100.000 *Bakeries* computed by use of the algorithms *Uniform Selection Hill Climber*, *Weighted Selection Hill Climber*, *GRASP Random* and *GRASP Sorted*. The agility of the solutions is illustrated by green, red, cyan and magenta lines which is the accumulated consumption of the first, second, third and fourth quarter of units, respectively, when the portfolio is sorted according agility as specified by Remark 1.

Power Plant Dispatch Problem with Agility for mixed portfolios. Two of the methods are so-called pure methods, denoted *Agile-Balancing* and *Predictive-Balancing* and these methods were first presented in [3]. In the simulation example presented in [3] it is illustrated that *Agile-Balancing* can out perform *Predictive-Balancing* both in terms of computation time and solution quality. A more systematic analysis (see Table 3.9) has, however, revealed that on average the solution quality obtained by *Agile-Balancing* is only slightly better than that of *Predictive-Balancing*. The pure methods thus illustrate that although prediction and agility are both important concepts to consider, when developing heuristic dispatch algorithms, neither one is sufficient in itself. Therefore two so-called hybrid methods are presented, which propose two very different ways of combining the advantages of prediction and agility. The hybrid methods are denoted *Predictive-Balancing-with-Agility* and *Agile-Balancing-with-Prediction* and they have not been previously published in [3].

Predictive-Balancing and *Predictive-Balancing-with-Agility* will be presented first. These methods are moving horizon approaches where a solution of the Virtual Power Plant Dispatch Problem is found by solving a series of smaller optimization problems. In later simulations CPLEX [65] will be used to solve sub-problems. The alternatives *Agile-Balancing* and *Agile-Balancing-with-Prediction* have been developed specifically for speed and are therefore based on sorting rather than solving optimization problems.

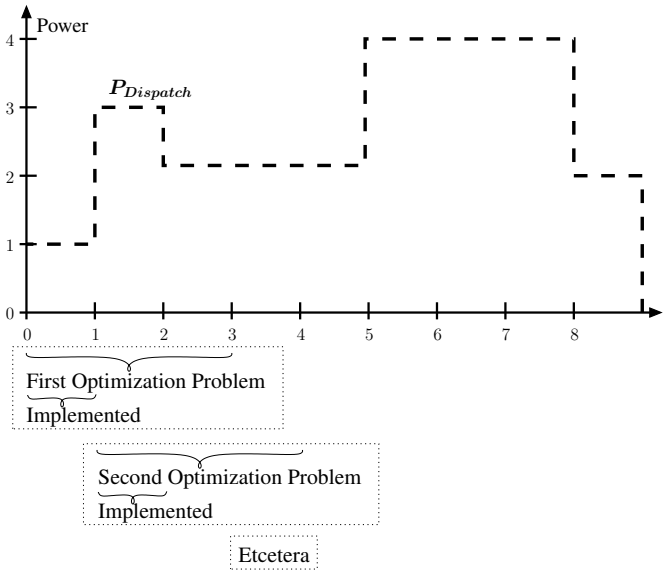

Figure 3.29: Moving horizon approach of *Predictive-Balancing* and *Predictive-Balancing-with-Agility* for a prediction horizon $K_{Prediction} = 3$ and an implementation horizon of one sample.

Predictive-Balancing and Predictive-Balancing-with-Agility

The algorithms *Predictive-Balancing* and *Predictive-Balancing-with-Agility* are moving horizon approaches and the difference between the two methods is that they solve the Virtual Power Plant Dispatch Problem and the Virtual Power Plant Dispatch Problem with Agility respectively.

To determine a solution of problem (3.16) to (3.17) *Predictive-Balancing* is given perfect prediction of $P_{Dispatch}$ over a certain prediction horizon $K_{Prediction} < K$, and the following optimization problem is solved:

$$\min_{P_i(\cdot)} \quad \sum_{k=1}^{K_{Prediction}} w_k |\mathcal{S}(k)| \tag{3.25}$$

$$s.t.$$

$$\sum_{i=1}^{N} P_i(k) + \mathcal{S}(k) = P_{Dispatch}(k), \tag{3.26}$$

where $w_{k_1} > w_{k_2}$ if $k_1 < k_2$. Remember that adding the impatience weights, w_k, to the cost function ensures that if the problem cannot be solved without introducing slack, then the imbalances will appear as late on the prediction horizon as possible. The Virtual Power Plant Dispatch Problem is solve by using moving horizon approach, so after solving problem (3.25) to (3.26) the results for the first sample is implemented and the optimization problem is progressed one sample (see Figure 3.29).

The method *Predictive-Balancing* does not consider agility in any way, which means that it will often run into the problem of having a to short prediction horizon as described in Section 3.3. This will also be demonstrated in the simulations below. Agility is therefore added to *Predictive-Balancing* in order to arrive at *Predictive-Balancing-with-Agility*. To do this agility weights, $w_{i,k}$, are introduced as described in Section 3.6. Again perfect prediction of $P_{Dispatch}$ is given over a certain prediction horizon $K_{Prediction} < K$, and the following optimization problem is solved:

$$\min_{P_i(\cdot)} \sum_{k=1}^{K_{Prediction}} \left(w_k |\mathcal{S}(k)| + \sum_{i=1}^{N} w_{i,k} |P_i(k)| \right) \tag{3.27}$$

$$s.t.$$

$$\sum_{i=1}^{N} P_i(k) + \mathcal{S}(k) = P_{Dispatch}(k). \tag{3.28}$$

Using the moving horizon approach the results for the first sample is implemented and problem (3.27) to (3.28) is progressed one sample (see Figure 3.29).

Agile-Balancing and Agile-Balancing-with-Prediction

The algorithms *Agile-Balancing* and *Agile-Balancing-with-Prediction* have been specifically designed for speed of computation and therefore uses sorting based approaches to determine a solution of the Virtual Power Plant Dispatch Problem with Agility.

The algorithm *Agile-Balancing* is based on the idea of agility maximization, where the worst quality units are dispatched at each sample. The idea is simple: At each sample the algorithm will first focus on the assignments of charging *Batteries* and starting *Bakeries*. The most pressing task is solved first and the unit with the smallest Agility Factor is the most critical asset in need of service. At sample k *Agile-Balancing* therefore dispatches as much power as possible to the *Batteries* and *Bakeries*, but no more than $P_{Dispatch}(k)$. Secondly, *Agile-Balancing* uses the buffer available in the *Buckets* to minimize any remaining imbalance.

Since there are no energy requirements on a *Bucket*, it can only constitute a resource and never a constraint. There are, however, both power and energy constraints on the *Bucket*, so only a limited amount of power can be dispatched to the *Bucket*-portion of the portfolio at each sample. The maximum amount of power, which can be dispatched to a portfolio containing $N^{Buckets}$ *Buckets* is therefore

$$P^{Bucket}_{Reserve}(k) = \sum_{i=1}^{N^{Buckets}} \min \left(\overline{P}_i, \frac{\overline{E}_i - E_i(k)}{T_s} \right).$$

Furthermore, *Agile-Balancing* handles any dispatch to *Buckets* by implementing the linear cost function given in [2]. Pseudo-code for *Agile-Balancing* is given in Algorithm 1.

As discussed in Section 3.6 a *Battery* or a *Bakery* can go from being a flexible resource to becoming a constraint as the deadline, K_{End}, approaches. Forced consumption on *Battery*$_i$ or *Bakery*$_i$ at sample k can be computed based on the Agility Factors introduced

in Section 3.3, as

$$P^{Battery}_{Forced,i}(k) = \begin{cases} 0 & \mathcal{K}^{Battery}_i > 1 \\ \overline{P}_i(1 - \mathcal{K}^{Battery}_i) & 1 \geq \mathcal{K}^{Battery}_i > 0 \\ \overline{P}_i & \mathcal{K}^{Battery}_i = 0 \end{cases}$$

and

$$P^{Bakery}_{Forced,i}(k) = \begin{cases} 0 & \mathcal{K}^{Bakery}_i > 1 \\ \overline{P}_i & \mathcal{K}^{Bakery}_i = 0, \end{cases}$$

and these requirements are of course also respected by the algorithm.

Algorithm 1:
Agile Balancing $\Big(\{LU_i\}_{i=1,2,...,N}, P_{Dispatch} \Big)$

for $k = 1$ **to** K **do**
  $P_{Forced}(k) = \sum_{i=1}^{N^{Batteries}} P^{Batteries}_{Forced,i}(k) + \sum_{j=1}^{N^{Bakeries}} P^{Bakeries}_{Forced,j}(k)$.
  if $P_{Forced}(k) > P_{Dispatch}(k)$ **then**
    $P^{Batteries}(k) = P^{Batteries}_{Forced}(k)$,
    $P^{Bakeries}(k) = P^{Bakeries}_{Forced}(k)$.
  else
    Sort *Batteries* and *Bakeries* according to increasing Agility Factors.
    Distribute $P_{Dispatch}(k)$ to *Batteries* and *Bakeries* in increasing Agility Factor order and such that $P^{Batteries}(k) + P^{Bakeries}(k)$ is as large as possible, but less than or equal to $P_{Dispatch}(k)$.
  end if
  $P^{Buckets}(k) = \min\Big(P^{Buckets}_{Reserve}(k), P_{Dispatch}(k) - P^{Batteries}(k) - P^{Bakeries}(k) \Big)$.
  Distribute $P^{Buckets}(k)$ to the *Buckets* in decreasing agility Agility Factor order, [2].
  $S(k) = P_{Dispatch}(k) - P^{Buckets}(k) - P^{Batteries}(k) - P^{Bakeries}(k)$.
end for

The method *Agile-Balancing* is very computationally efficient, but it has one major flaw: Since the algorithm is strictly non-predictive it can sometimes start too many *Bakeries* at times when a drop in power is closely pending. If the *Bucket* buffer is not sufficiently large, then this can cause significant slack (see the Results section below). To remedy this problem the element of prediction is introduced to arrive at *Agile-Balancing-with-Prediction*.

Prediction is added to *Agile-Balancing* in the form of a prediction test, which is performed before starting each *Bakery*. In the prediction test it is checked whether starting the current *Bakery* in combination with all other running units will cause imbalances in the next $K_{Prediction}$ samples. If this is the case, then *Agile-Balancing-with-Prediction* will not start this *Bakery* just yet, but move on to the next unit.

The prediction test means that occasionally *Bakeries* are skipped even though they have the same Agility Factor as one or more *Batteries*. It is therefore necessary to give *Bakeries* slightly higher priority than *Batteries* throughout the simulations. A scaling factor $\alpha < 1$ is therefore introduced and *Batteries* and *Bakeries* are sorted according to $\mathcal{K}^{Batteries}$ and $\alpha\mathcal{K}^{Bakeries}$. This insures that *Bakeries* are given higher priority than *Batteries*

when running the algorithm and balances out the prediction test. Again to test the algorithm it must first be properly tuned to find the appropriate value of α. This is done in the results section below.

Pseudo-code for *Agile-Balancing-with-Prediction* is given in **Algorithm 2** and **3**.

Algorithm 2:
Agile-Balancing-with-Prediction$\Big(\{LU_i\}_{i=1,2,\ldots,N}, \{P_{Dispatch,k}\}_{k=1,2,\ldots,K}, K_{Prediction}, \alpha\Big)$

*Pseudo-code is the same as **Algorithm 1** except insert the following initialization before main **for** loop*

1: Define $\{P_{Bakeries}\}_{k=1,2,\ldots,K}$ as array of zeros of length K.
also replace line 7 with
2: Compute $\mathcal{K}^{Batteries}$ and $\alpha\mathcal{K}^{Bakeries}$ and sort *Batteries* and *Bakeries* according to these factors.
and line 8 with
3: Distribute $P_{Dispatch}(k)$ to *Batteries* and *Bakeries* in sorted order and such that $P^{Batteries}(k)+P^{Bakeries}(k)$ is as large as possible, but less than or equal to $P_{Dispatch}(k)$. However start each *Bakery* only if **Prediction-Test**(*Bakery*, k, $K_{Prediction}$, $P_{Bakeries}$, $P_{Dispatch}$) is true.

Algorithm 3:
Prediction-Test($Bakery, k, K_{Prediction}, P_{Bakeries}, P_{Dispatch}$)

for $m = 1$ **to** $K_{Prediction}$ **do**
  if $P_{Dispatch}(k+m) - P_{Bakeries}(k+m) - \overline{P}_{Bakery} < 0$ **then**
    return false
  end if
end for
for $n = 1$ **to** $K_{Run,Bakery}$ **do**
  $P_{Bakeries}(k+n) = P_{Bakeries}(k+n) + \overline{P}_{Bakery}$
end for
return true

Results

In the following simulation example a randomly generated portfolio of 105 units is considered with $N^{Buckets} = 5$ and $N^{Batteries} = N^{Bakeries} = 50$. All units have $\frac{\overline{E}}{T_s\overline{P}} \leq 10$ and $\sum_{Portfolio} \overline{E} = 50$. In all simulations $T_s = 1$ and $E_{i,0} = 0$ for all units. Solutions of problems (3.25) to (3.26) and (3.27) to (3.28) are computed by use of CPLEX, [65]. *Agile-Balancing* and *Agile-Balancing-with-Prediction* have been implemented in C#, [66]. Computations are performed on a standard laptop PC.

To compare algorithms *Agile-Balancing-with-Prediction* must first be tuned to find a suitable value of the parameter α. Tuning is performed on a separate training set of 100 problem instances with the same structure as described above. The parameter α is tested for the values $\frac{1}{2}, \frac{2}{3}, \frac{3}{4}, \frac{4}{5}$ and the result of the parameter tuning test is that α should be $\frac{2}{3}$ for this problem structure.

The results of running *Predictive-Balancing* for $K_{Prediction} = 10$ are given in Figure 3.30. It can be seen that the utilization of the portfolio is very erratic, and also

considerable slack is introduced around sample 25 and 50. When there is a drop in $P_{Dispatch}$, *Predictive-Balancing* attempts to use the *Buckets* as buffer to maintain the balance between supply and demand. Towards the end of each low power period, however, *Predictive-Balancing* has significant slack. This is of course due to the prediction horizon not being sufficiently long, and the problem could be mended by increasing the prediction horizon. This modification, however, comes at the price of computation time.

The results of running *Agile-Balancing* on the considered portfolio is given in Figure 3.31. Again considerable slack is introduced during the second low power period, but now there is a very different explanation: Since *Agile-Balancing* is strictly non-predictive it has no information about the sudden drop in power at sample 40. It therefore starts a significant number of *Bakeries* just before that time. When the power drops, the *Battery* portion of the portfolio can be turned down, but the *Bakeries* cannot. *Agile-Balancing* also attempts to use the *Buckets* as buffer to maintain the balance between supply and demand, but the reservoir is not large enough to achieve this goal.

The results of running *Predictive-Balancing-with-Agility* and *Agile-Balancing-with-Prediction* for $K_{Prediction} = 10$ and $\alpha = \frac{2}{3}$ are given in Figure 3.32 and 3.33 respectively. Inspecting Figure 3.32 and 3.33 it can be seen that both hybrid methods generate very good solutions of the considered dispatch problem, as neither method have to introduce any slack. Towards the end of the first low power period *Agile-Balancing-with-Prediction* has to start some *Batteries* due to approaching deadlines, but as it is well prepared, there is enough buffer in the *Bucket* portion of the portfolio to still maintain balance between supply and demand.

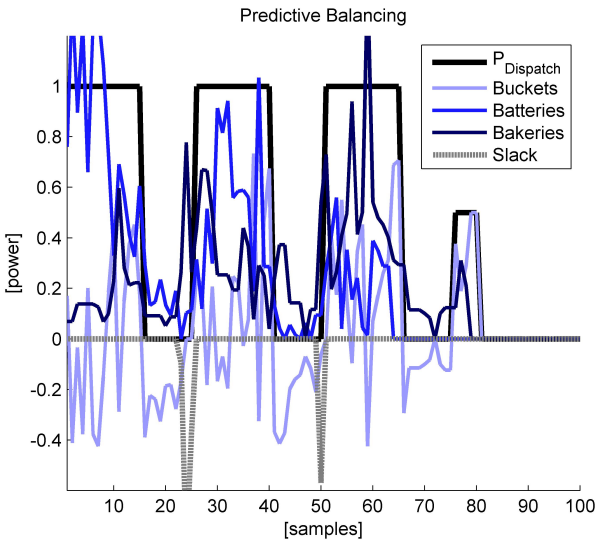

Figure 3.30: Power dispatched at each sample for each type of unit by *Predictive-Balancing* when $K = 10$.

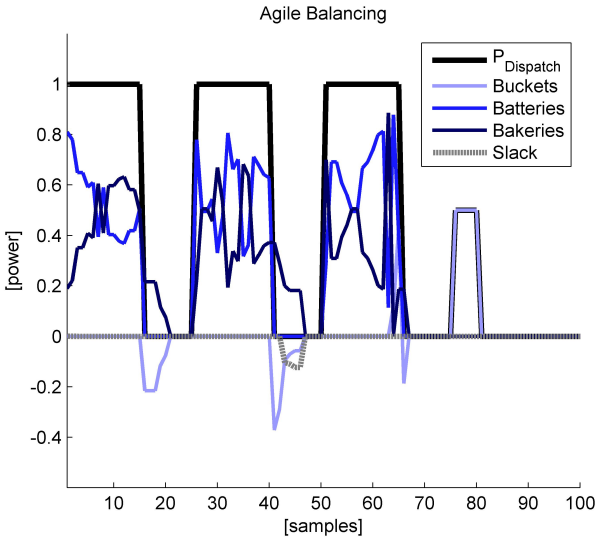

Figure 3.31: Power dispatched at each sample for each type of unit by *Agile-Balancing*.

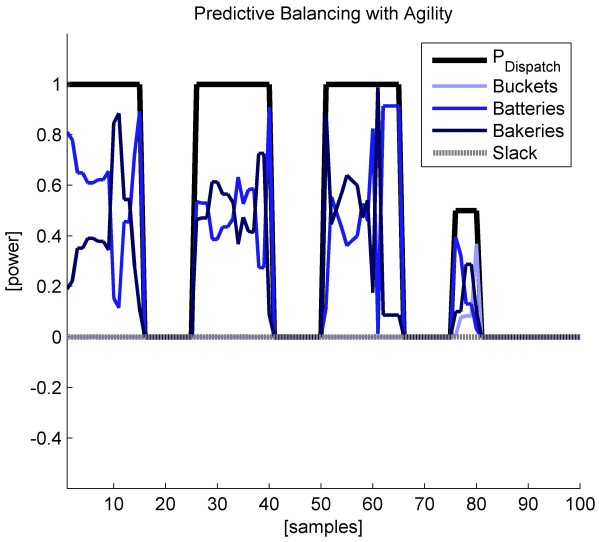

Figure 3.32: Power dispatched at each sample for each type of unit by *Predictive-Balancing-with-Agility* when $K = 10$.

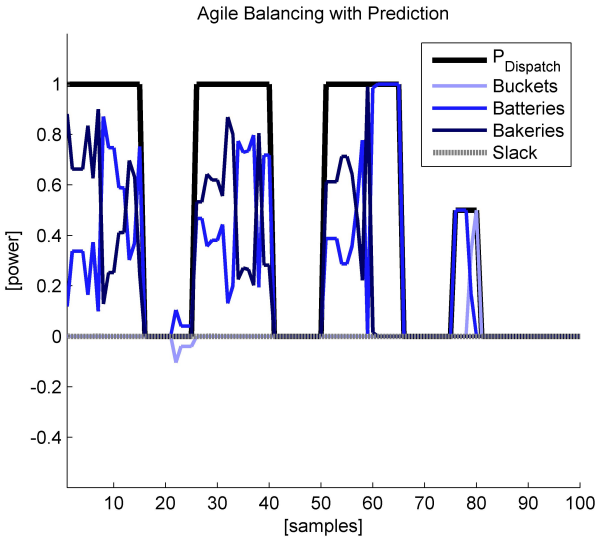

Figure 3.33: Power dispatched at each sample for each type of unit by *Agile-Balancing-with-Prediction*.

In the next simulation example computations are repeated for 100 randomly generated portfolios each of 105 units, where $N^{Buckets} = 5$ and $N^{Batteries} = N^{Bakeries} = 50$. Again all units have $\frac{\overline{E}}{T_s \overline{P}} \leq 10$ and $\sum_{Portfolio} \overline{E} = 50$. The results are presented in Figure 3.34 and Table 3.9.

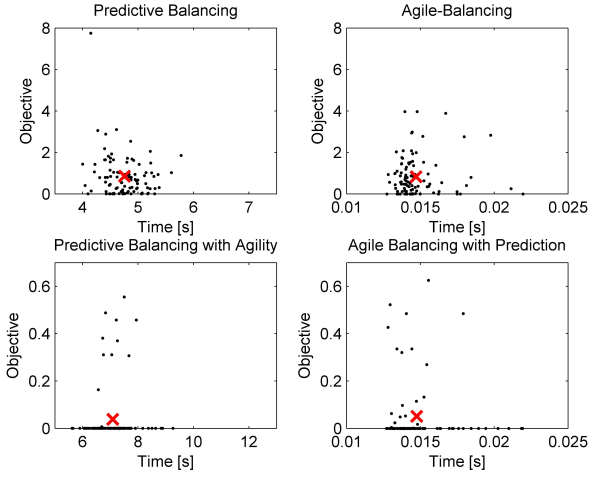

Figure 3.34: Results of 100 runs of *Predictive-Balancing*, *Agile-Balancing*, *Predictive-Balancing-with-Agility* and *Agile-Balancing-with-Prediction*. Red crosses are the geometric mean. Axis scales vary.

From Table 3.9 it is clear that the best results by far are obtained by the hybrid methods, with *Predictive-Balancing-with-Agility* out-performing *Agile-Balancing-with-Prediction* with about 13%. This illustrates that both agility and prediction must be considered when designing dispatch methods. Table 3.9 also shows that the sorting-based algorithms are much faster at finding a solution than the moving horizon methods. This means that the average 13% improvement in objective cost from *Predictive-Balancing-with-Agility* to *Agile-Balancing-with-Prediction* takes more than 700 times as long to compute.

Notice also that although *Agile-Balancing-with-Prediction* has to do the Prediction-Check in **Algorithm 3**, the average computation time is the same as that of *Agile-Balancing*. For *Predictive-Balancing* and *Predictive-Balancing-with-Agility*, however, adding agility gives an increase in average computation time from 4.76 seconds to 7.12 seconds.

	Comp. Time [s]	$\sum \lvert\mathcal{S}(\cdot)\rvert$
Predictive-Balancing	4.76	0.8437
Agile-Balancing	0.01	0.8156
Predictive-Balancing-with-Agility	7.12	0.0381
Agile-Balancing-with-Prediction	0.01	0.0436

Table 3.9: Computation time and the average sum of numerical imbalances for 100 runs of *Predictive-Balancing*, *Agile-Balancing*, *Predictive-Balancing-with-Agility* and *Agile-Balancing-with-Prediction*.

4 | Closing Remarks

This Thesis has investigated some of the challenges relating to the development of Virtual Power Plants with special attention to flexibility, value creation and computational complexity of portfolio coordination. In this section the main conclusions are summarized based on Research Questions 1 to 3 as formulated in Section 1.2. After this perspectives on future work are given.

4.1 Conclusions

Our first contribution was given in Section 3.1 with the introduction of the *Buckets*, *Batteries and Bakeries* taxonomy for modelling flexibility. Although the archetypal flexibility categories are very simple and not novel as such (see Table 2.1), hopefully we have demonstrated throughout Section 3 that having a short and concise taxonomy for defining flexibility is very useful. In a way the strength of the taxonomy is that it provides a vocabulary for discussing flexibility, see [33], [72] and [73]. It is simply far more practical to say "the *Battery*" than have to state "a power and energy constraint integrator with charge constraint at user specified deadline" over and over again. We have also shown, that though simple the models are sufficiently complex to capture and illustrate many interesting points and issues relating to agility, value creation and dispatch methods.

Research Question 1: Is a better quality of flexibility also more valuable?

To investigate Research Question 1 a three stage market model was developed in Section 3.2. The model includes the Day-Ahead Market, the Intra-Day Market and the Regulating Power Market. Historic prices from the Nordic electricity markets and price zones DK1 and DK2 were used in calculations. To capture flexibility of different quality the considered portfolio consisted of one unit of each type in the taxonomy. The same parameter values were used for all three types of units, which made it possible to directly compare the effects that different constraints have on the revenue potential.

It was found that the revenue potential is highly dependent on the quality of flexibility. The fixed energy requirement of the *Battery* and *Bakery* meant that the full costs of the baseline energy purchase could not be recovered. Still, however, it was shown that significant savings can be achieved from trading across several markets. Specifically it was found that the *Battery* can earn 14% of the total profit earned by the *Bucket* and the *Bakery* can earn 6% of that profit. This confirms the hypothesis that a better quality of flexibility is also more valuable.

Research Question 2: How can the Virtual Power Plant preserve the quality of the flexibility in the portfolio as flexible units are dispatched?

In Section 3.3 the concept of agility was introduced in order to investigate Research Question 2. Agility means to maximize the quality of flexibility in the portfolio as flexible units are dispatched. It was demonstrated that agility can make the Virtual Power Plant more robust in situations

1. when predictions of $P_{Dispach}$ are erroneousness, and
2. when the length of the prediction horizon of $P_{Dispatch}$ is not sufficient.

The increased robustness is a result of the fact that agility generates more manoeuvrability or a larger solution space for dispatch problems to be solved in future time steps.

Agility Factors were introduced in Section 3.3 to illustrate how flexibility quality can be preserved as units are dispatched. The Agility Factor of a unit reflects the flexibility quality of the unit and through Agility Factors an ordering by quality of the flexible units in a portfolio can be achieved. In Section 3.6 it was demonstrated how to build Agility Factors into the Virtual Power Plant Dispatch Problem formulation to find solutions, which minimizes slack, but are also agile. By solving the Virtual Power Plant Dispatch Problem with Agility it is therefore possible to preserve the quality of the flexibility in the portfolio as units are dispatched.

Research Question 3: Is coordination of a large portfolio of flexible units computationally challenging for the Virtual Power Plant? How can this issue be mitigated?

Research Question 3 has been addressed through analytical contributions, by investigating exact methods for finding solutions of the Virtual Power Plant Dispatch Problem and by investigating heuristic optimization:

Analytical contributions relating to the Virtual Power Plant Dispatch Problem and the *Buckets*, *Batteries and Bakeries* taxonomy were given in Section 3.4. It was proven that optimal, causal dispatch strategies do not generally exist for a portfolio of *Buckets*, *Batteries and Bakeries*. In the special case of a portfolio of only *Buckets* having $\underline{P} = \underline{E} = 0$, however, an optimal, causal strategy does exist. This strategy can be found by solving a quadratic problem at each sample. It was also proven that when the portfolio contains *Bakeries* the Virtual Power Plant Dispatch Problem is NP-complete. This means that the computation time associated with finding an optimal solution using currently known algorithms grows extremely fast with the problem size.

In Section 3.5 several attempts were made to determine optimal solutions of larger problem instances of the Virtual Power Plant Dispatch Problem. Specifically the software package CPLEX was used and *Dynamic Programming* and *Dantzig-Wolfe* Decomposition was investigated. For CPLEX and *Dynamic Programming* is was found that the methods could successfully solve the Virtual Power Plant Dispatch Problem, but did not scale to very large problem instances ($>$50 units). This is because the introduction of *Bakeries* to the portfolio makes the dispatch problem mixed integer/combinatorial in nature. For *Dantzig-Wolfe* Decomposition the method was inapplicable due to a more structural issue: As iterations progresses the Master Problem can generate convex combinations of

Bakery consumption plans, which are not feasible. The method will then terminate with an infeasible solution even though the Sub-Problem does respect the run time constraints of *Bakeries*.

In Section 3.7 the option of solving the Virtual Power Plant Dispatch Problem by use of heuristic optimization was investigated.

In initial efforts *Hill Climber* and *GRASP* (Greedy Randomized Adaptive Search Procedure) were adapted to solve the Virtual Power Plant Dispatch Problem for a portfolio of *Bakeries*. Four algorithms were developed, denoted *Uniform Selection Hill Climber*, *Weighted Selection Hill Climber*, *GRASP Random* and *GRASP Sorted*. After tuning and testing, by far the best results where obtained by *GRASP Sorted*. This method can determine solutions, which are both agile and have very little slack even for problems of 100.000 units and 100 samples with a computation time of just 10 seconds.

Next the objective was to investigate the relationship between prediction and agility further and develop heuristic optimization methods specifically designed to have short computation times. Four methods were presented: Two pure methods and two hybrid methods. The pure methods were denoted *Predictive-Balancing* and *Agile-Balancing*. *Predictive-Balancing* has perfect prediction of $P_{Dispatch}$ over a certain horizon, but does not consider agility in any way. *Agile-Balancing*, on the other hand, is strictly non-predictive and generates a dispatch based only on agility information about the portfolio and the current value of $P_{Dispatch}$. It was found through simulations, that the performance of the two methods is comparable in terms of solution quality. The computation time for *Agile-Balancing* is however nearly 500 times less than that of *Predictive-Balancing* for the considered set-up.

The pure methods illustrated that although prediction and agility are both important concepts to consider when developing heuristic optimization algorithms neither is sufficient in itself. Consequently two hybrid methods were developed, since the hybrid methods propose two very different ways of combining the advantages of prediction and agility. The first hybrid method is denoted *Predictive-Balancing-with-Agility*. Like *Predictive-Balancing* this method generates a dispatch based on perfect prediction of $P_{Dispatch}$ over a certain prediction horizon, but Agility Factors are also added to the formulation of the Virtual Power Plant Dispatch Problem. The second hybrid method is denoted *Agile-Balancing-with-Prediction*. Like *Agile-Balancing* this method generates a dispatch based on agility information about the portfolio, but it also performs a prediction test to foresee rapid fluctuations of $P_{Dispatch}$ in the near future. It was again found through simulations that the performance of the two hybrid methods is comparable in terms of solution quality. The computation time for *Agile-Balancing-with-Prediction* is, however, 700 times less than that of *Predictive-Balancing-with-Agility* for the considered set-up.

4.2 Future Work

The work presented in this Thesis has investigated Research Questions 1 to 3 as formulated in Section 1.2. There are, however, several ways to extend and elaborate the presented concepts and ideas. Suggestions for such future work is given below.

Flexibility

The presented taxonomy could be extended with several other models to include constraints such as ramp rates, activation time or multi-phase batch processes. With this extension the hierarchy of flexibility quality would most likely be lost. It might however be possible to formulate Agility Factor type values based on statistical analysis of the portfolio and feasible progressions of $P_{Dispatch}$. Such estimations would probably be highly problem specific, but a general methodology might be obtained.

A different extension of the propose flexibility modelling is to evaluate to what extent the utilization of simplified models at the aggregator level leads to reasonable utilization of real systems. Certainly flexibility models such as *Buckets*, *Batteries and Bakeries* do not capture all dynamics and features of a real system. A continuous update of parameter values in the simple aggregator models based on feedback from the actual system could however lead to at better utilisation of actual systems. This would of course require fast and reliable two way communication, but it would not complicate the computational complexity at the level of the Virtual Power Plant. This line of work has already begun with [74].

Value Creation

An interesting extension of the presented set-up would be to introduce several Virtual Power Plants, which are competing in a market environment similar to the one presented in Section 3.2. To do so, however, it must first be determined how prices should be settled on those markets to achieve some form of feedback in the formation of prices. It could then be investigated how well agility is obtained "system wide" if each Virtual Power Plant is individually maximizing its own flexibility quality. It would also be interesting to see whether "system wide agility" is dependent on how units are distributed among the Virtual Power Plants. It could then be examined whether it is better that each Virtual Power Plant has a mixed portfolio of units or if similar units should be grouped in a designated Virtual Power Plant.

Portfolio Coordination

The presented dispatch problem could be extended with geographical information by including a power grid in the problem formulation. The cost function should then include balancing, grid congestion management and agility. If an effective way of included the grid could be encoded, then for the exact optimization methods the seeming complication of the problem given by including a grid could actually be a blessing in disguise. This is because many combinations of starting *Bakeries* would become infeasible due to violation of grid constraints. The sheer number of feasible combination is the main challenges for e.g. the *Dynamic Programming* algorithm presented in Section 3.5, so it can be expected that much larger problem instances could be solved within a reasonable computation time.

It would also be interesting to see whether the sorting based methods could be adapted to include grid limitations such that computation times comparable to the ones seen for *Agile-Balancing* and *Agile-Balancing-with-Prediction* could still be obtained.

References

[1] Mette Kirschmeyer Petersen, Lars Henrik Hansen and Tommy Mølbak, "Exploring the Value of Flexibility: A Smart Grid Discussion", *IFAC Symposium on Power Plants and Power System Control*, vol. 8, no. 1, pp. 43–48, 2012.

[2] Mette Kirschmeyer Petersen, Jan Dimon Bendtsen and Jakob Stoustrup, "Optimal dispatch strategy for the Agile Virtual Power Plant", *American Control Conference*, pp. 288–294, 2012.

[3] Mette Kirschmeyer Petersen, Kristian Edlund, Lars Henrik Hansen, Jan Dimon Bendtsen and Jakob Stoustrup, "A Taxonomy for Modeling Flexibility and a Computationally Efficient Algorithm for Dispatch in Smart Grids", *American Control Conference*, pp. 1150–1156, 2013.

[4] Mette Kirschmeyer Petersen, Lars Henrik Hansen, Jan Dimon Bendtsen, Kristian Edlund and Jakob Stoustrup, "Market integration of Virtual Power Plants", *IEEE Conference on Decision & Control*, pp. 2319–2325, 2013.

[5] Mette Kirschmeyer Petersen, Lars Henrik Hansen, Jan Dimon Bendtsen, Kristian Edlund and Jakob Stoustrup, "Heuristic Optimization for the Discrete Virtual Power Plant Dispatch Problem", *IEEE Transactions on Smart Grid, Vol. 5, No. 6, pp. 2910–2918, 2014*.

[6] Klaus Trangbaek, Mette Kirschmeyer Petersen, Jan Dimon Bendtsen and Jakob Stoustrup, "Exact Power constraints in Smart Grid Control", *IEEE Conference on Decision & Control*, pp. 6907–6912, 2011.

[7] Klaus Trangbaek, Mette Kirschmeyer Petersen, Jan Dimon Bendtsen and Jakob Stoustrup,, "Predictive Smart Grid Control with Exact Aggregated Power Constraints", *Smart Power Grids 2011*, Springer Berlin Heidelberg, pp. 655–674, 2012.

[8] P. Ø. Andreasen and L. Aagaard, *Smart Grid i Danmark*, Energinet.dk and Dansk Energi, Report, 2010.

[9] Jack A. Nickerson and Todd R. Zenger, *Envy, Comparison Costs , and the Economic Theory of the Firm*, Strategic Manament Journal, Volume 29, 2008, pp. 1429-1449.

[10] D. Pudjianto, C. Ramsay and G. Strbac, *Virtual Power Plant and System Integration of Distributed Energy Resources*, Renewable Power Generation, IET, Volume: 1, Issue: 1, pp. 10–16, 2007.

[11] S. Harbo and B. Biegel, *Contracting Flexibility Services*, 4th IEEE PES Innovative Smart Grid Technologies Europe, pp. 1–5, 2013.

[12] Energinet.dk, *Systemydelser til levering i Danmark, Udbudsbetingelser*, 2012.

[13] Steven Stoft (2002), *Power System Economics Designing Markets for Electricity*, Wiley-IEEE Press 1 (1) p. 63-64.

[14] Kristian Edlund, Jan Dimon Bendtsen and Tommy Mølbak, *Simple Models for Model-based Portfolio Balancing Controller Sythesis*, IFAC Symposium on Power Plants and Power Systems Control, 2009.

[15] Tobias Gybel Hovgaard, Lars Finn Sloth Larsen and John Bagterp Jørgensen (2011), *Robust Economic MPC for a Power Management Scenario with Uncertainties*, 50th IEEE Conference on Decision and Control and European Control Conference, Orlando, December 2011, pp. 1515-1520.

[16] Shi You, Chresten Treaholt and Bjarne Poulsen (2009), *A Study on Electricity Export Capability of the μCHP System with Spot Price*, IEEE Power & Energy Society General Meeting, 2009, pp. 1-6

[17] Fatemeh Tahersima, Jakob Stoustrup, Soroush Afkhami Meybodi, and Henrik Rasmussen, *Contribution of Domestic Heating Systems to Smart Grid Control*, 50th IEEE Conference on Decision and Control and European Control Conference, Orlando, December 2011, pp. 3677-3681.

[18] Wangensteen, I., Power System Economics - the Nordic Electricity Market, *Tapir Academic Press*, 2007.

[19] K. Edlund, J.D. Bendtsen and J.B. Jørgensen, *Hierarchical model-based predictive control of a power plant portfolio*, Control Engineering Practice, Vol. 19, pp. 1126–1136, 2011.

[20] K. Aoki, T. Satoh, M. Itoh, T. Ichimori, and K. Masegi, *Unit Commitment in a Large-Scale Power System including Fuel Constrained Thermal and Pumped-Storage Hydro*, IEEE Transactions on Power Systems, Vol 2(4), pp. 1077–1084, 1987.

[21] A. K. Ayuob and A. D. Patton, *Optimal thermal generating unit commitment*, IEEE Transactions on Power Apparatus and Systems, 90(4):1752–1756, 1971.

[22] A. P. Houmøller, *The Nordic Electricity Exchange and the Nordic Model for a Liberalised Electricity Market*, Business Report, www.nordpoolspot.com, 2009.

[23] C. Jørgensen, J. H. Mortensen, Tommy Mølbak and E. O. Nielsen, *Modelbased Fleet Optimization and Master Control of a Power Production System*, IFAC Symposium on Power Plants and Power Systems Control, pp. 201–206, 2006.

[24] T. Kudela, L. Sukhodolska, L. L. Lyck, *The Nordic Electricity Market including Danish Ancillary Services*, Internal DONG Energy report, 2012.

[25] K. Heussen, S. You, B. Biegel, L. H. Hansen and K. B. Andersen, *Indirect Control for Demand Side Management A Conceptual Introduction*, 3rd IEEE PES Innovative Smart Grid Technologies Europe, pp. 1–8, 2012.

[26] Shi You, Chresten Treaholt and Bjarne Poulsen, *A Market-based Virtual Power Plant*, International Conference on Clean Electrical Power Renewable Energy Resources Impact (ICCEP), pp.460–465, 2009.

[27] M. Roozbehani, A. Faghih, M.I. Ohannessian and M.A. Dahleh, *The Intertemporal Utility of Demand and Price Elasticity of Consumption in Power Grids with Shiftable Loads*, pp. 1539 - 1544, 50th IEEE Conference on Decision and Control and European Control Conference, 2011.

[28] P. Nyeng and J. Østergaard, *Information and Communications Systems for Control-by-Price of Distributed Energy Resources and Flexible Demand*, IEEE Transactions on Smart Grid, Vol. 2, No. 2, 2011.

[29] Duncan S. Callaway and Ian A. Hiskens, *Achieving Controllability of Electric Loads*, Proceedings of the IEEE, Vol. 99, No. 1, January 2011.

[30] M. Juelsgaard, P. Andersen, and R. Wisniewski, *Stability Concerns for Indirect Consumer Control in Smart Grids*, Proceedings of the European Control Conference, pp. 2006-2013, 2013.

[31] Benjamin Biegel, Palle Andersen, Jakob Stoustrup, Lars Henrik Hansen and David Victor Tackie, *Information Modelling for Direct Control of Distributed Energy Resources*, American Control Conference 2013, pp. 3498–3504, 2013.

[32] Jan Bendtsen, Klaus Trangbaek and Jakob Stoustrup, *Hierarchical Model Predictive Control for Resource Distribution*, Conference on Decision and Control 2010, pp. 2468–2473, 2010.

[33] Samira Rahnama, Jakob Stoustrup and Henrik Rasmussen, *Integration of Heterogeneous Industrial Consumers to Provide Regulating Power to the Smart Grid*, Conference on Decision and Control 2013, pp. 6268–6273, 2013.

[34] Samira Rahnama, Jakob Stoustrup and Henrik Rasmussen, *Model Predictive Control for Integration of Industrial Consumers to the Smart Grid under a Direct Control Policy*, 2013 IEEE International Conference on Control Applications, pp. 515–520, 2013.

[35] Luminita C. Totu, John Leth and Rafael Wisniewski, *Control for large scale demand response of thermostatic loads*, 2013 American Control Conference, pp. 5023–5027, 2013

[36] Kai Heussen, Stephan Koch, Andreas Ulbig, Göran Andersson, *Energy Storage in Power System Operation: The Power Nodes Modeling Framework*, IEEE PES Conference on Innovative Smart Grid Technologies Europe, 2010, pp. 1-8.

[37] Benjamin Biegel, Jakob Stoustrup, Jan Bendtsen and Palle Andersen, *Model Predictive Control for Power Flows in Networks with Limited Capacity*, 2012 American Control Conference, 2012, pp. 2959-2964.

[38] Ali Faghih, Mardavij Roozbehani and Munther A. Dahleh, *Optimal Utilization of Storage and the Induced Price Elasticity of Demand in the Presence of Ramp Constraints*, 50th IEEE Conference on Decision and Control and European Control Conference, 2011, pp. 842-847.

[39] T.Y. Lee and N. Chen, *Effect of the Battery Energy Storage System on the Time Of Use Rates Industrial Customers*, IEE Proc.-Gener. Transm. Distrib., Vol 141, No. 5, September 1994, pp. 521-528.

[40] Matt Kraning, Yang Wang, Ekine Akuiyibo, Stephen Boyd, *Operation and Configuration of a Storage Portfolio via Convex Optimization*, 18th IFAC World Congress, 2011, pp. 10487-10492.

[41] Nikolaos Gatsis and Georgios B. Giannakis, *Residential Demand Response with Interruptible Tasks: Duality and Algorithms*, 50th IEEE Conference on Decision and Control and European Control Conference, 2011, pp. 1-6.

[42] Ioannis Ch. Paschalidis, Binbin Li, Michael C. Caramanis, *A Market-Based Mechanism for Providing Demand-Side Regulation Service Reserves*, 50th IEEE Conference on Decision and Control and European Control Conference, 2011, pp. 1-6., pp. 21-26.

[43] Konstantin Turitsyn, Scott Backhaus, Maxim Ananyev and Michael Chertkov, *Smart Finite State Devices: A Modeling Framework for Demand Response Technologies*, 50th IEEE Conference on Decision and Control and European Control Conference, 2011, pp. 7-14.

[44] B. Daryanian, R.E. Bohn and R.D. Tabors, *Optimal Demand-Side Response to Electricity Spot Prices for Storage-Type Customers*, IEEE Transactions on Power Systems, Vol. 4, No. 3, 1989, pp. 897-903.

[45] Amir-Hamed Mohsenian-Rad, Vincent W. S. Wong, Juri Jatskevich, Robert Schober and Alberto Leon-Garcia, *Autonomous Demand-Side Management Based on Game-Theoretic Energy Consumption Scheduling for the Future Smart Grid*, IEEE Transactions on Smart Grid, Vol. 1, No. 3, 2010, pp. 320-331.

[46] Angel Rosso, Juan Ma, Daniel S. Kirschen and Luis F. Ochoa, *Assessing the Contribution of Demand Side Management to Power System Flexibility*, 50th IEEE Conference on Decision and Control and European Control Conference, 2011, pp. 4361-4365.

[47] Changsun Ahn, Chiao-Ting Li and Huei Peng, *Decentralized Charging Algorithm for Electrified Vehicles Connected to Smart Grid*, American Control Conference, 2011, pp. 3924-3929.

[48] Anthony Papavasiliou and Shmuel S. Oren, *Supplying Renewable Energy to Deferrable Loads: Algorithms and Economic Analysis*, IEEE Power and Energy Society General Meeting, 2010, pp. 1-8.

[49] Ralph Hermans, Mads Almassalkhi and Ian Hiskens, *Incentive-based Coordinated Charging Control of Plug-in Electric Vehicles at the Distribution-Transformer Level*, 2012 American Control Conference, 2012, pp. 264-269.

[50] Amir-Hamed Mohsenian-Rad, Vincent W.S. Wong, Juri Jatskevich and Robert Schober, *Optimal and Autonomous Incentive-based Energy Consumption Scheduling Algorithm for Smart Grid*, Innovative Smart Grid Technologies Conference, 2010, pp. 1-6.

[51] A. Subramanianz, M. Garcia, A. Domnguez-Garca, D. Callawayx, K. Poollay and P. Varaiyay, *Real-time Scheduling of Deferrable Electric Loads*, 2012 American Control Conference, 2012, pp. 3643-3650.

[52] Jing Huang, Vijay Gupta and Yih-Fang Huang, *Scheduling Algorithms for PHEV Charging in Shared Parking Lots*, 2012 American Control Conference, 2012, pp. 276-281.

[53] Kin Cheong Sou, James Weimer, Henrik Sandberg, and Karl Henrik Johansson, *Scheduling Smart Home Appliances Using Mixed Integer Linear Programming*, 50th IEEE Conference on Decision and Control and European Control Conference, 2011, pp. 5144-5149.

[54] Shengbo Chen, Prasun Sinha and Ness B. Shroff, *Scheduling Heterogeneous Delay Tolerant Tasks in Smart Grid with Renewable Energy*, 51st IEEE Conference on Decision and Control, 2012, pp. 1130-1135,

[55] M. L. Putterman, *Markov Decision Processes, Discrete Stochastic Dynamic Programming*, Wiley-Interscience, 2005.

[56] D. T. Phan, J. Xiong and S. Ghosh, *A Distributed Scheme for Fair EV Charging Under Transmission Constraints*, American Control Conference, Montreal pp. 1053–1058, 2012.

[57] Z. A. Vale, H. Morais, H. Khodr, B. Canizes and J. Soares, *Technical and Economic Resources Management in Smart Grids using Heuristic Optimization Methods*, IEEE Power and Energy Society General Meeting, pp. 1–7, 2010.

[58] P. Faria, J. Soares, Z. Vale, H. Morais and T. Sousa, *Modified Particle Swarm Optimization Applied to Integrated Demand Response and DG Resources Scheduling*, IEEE Transactions on Smart Grid, Vol. 4, No. 1, pp. 606–616, March 2013.

[59] Z. Chen, L. Wu and Y. Fu, *Real-Time Price-Based Demand Response Management for Residential Appliances via Stochastic Optimization and Robust Optimization*, IEEE Transactions on Smart Grid, Vol. 3, No. 4, pp. 1822–1831, December 2012.

[60] T. Logenthiran, D. Srinivasan and T. Z. Shun, *Demand Side Management in Smart Grid Using Heuristic Optimization*, IEEE Transactions on Smart Grid, Vol. 3, No. 3, pp. 1244–1252, September 2012.

[61] S. Salinas, M. Li and P. Li, *Multi-Objective Optimal Energy Consumption Scheduling in Smart Grids*, IEEE Transactions on Smart Grid, Vol. 4, No. 1, pp. 341–348 March 2013

[62] P. Yi, X. Dong, A. Iwayemi, C. Zhou, and S. Li, *Real-Time Opportunistic Scheduling for Residential Demand Response*, IEEE Transactions on Smart Grid, Vol. 4, No. 1, pp. 227–234, March 2013.

[63] `http://www.energinet.dk/EN/El/Engrosmarked/Udtraek-af-markedsdata/Sider/default.aspx`

[64] Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein, *Introduction to Algorithms, Second Edition*, The MIT Press, 2001.

[65] `http://www-01.ibm.com/software/integration/optimization/cplex-optimization-studio/`

[66] `http://www.msdn.microsoft.com`

[67] F. Vanderbeck, *On Dantzig-Wolfe Decomposition in Integer Programming and Ways to Perform Branching in a Branch-And-Price Algorithm*, Operations Research, Vol. 48, No. 1., pp. 111-128, 2000.

[68] G. B. Dantzig and P. Wolfe, *Decomposition principle for linear programs*, Operations Research, 8(1), pp. 101-111.

[69] Edmund K. Burke and Graham Kendall, *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*, Springer, 2005.

[70] Thomas A. Feo and Maurico G.C. Resende, *Greedy Randomized Adaptive Search Procedures*, Journal of Global Optimization 1995, No. 6, pp. 109–133.

[71] Saul I. Gass and Carl M. Harris, *Encyclopedia of Operations Research and Management Science*, Springer US, 1996.

[72] R. Pedersen, J. Schwensen, S. Sivabalan, C. Corazzol, S. Shafiei, K. Vinther and J. Stoustrup, *Direct Control Implementation of a Refrigeration System in Smart Grid*, American Control Conference (ACC), pp. 3954–3959, 2013

[73] J.Hansen, J. Knudsen and M. Annaswamy, *Demand Response in Smart Grids: Participants, Challenges, and a Taxonomy*, available at web.mit.edu.

[74] Samira Rahnama, S. Ehsan Shafiei, Jakob Stoustrup, Henrik Rasmussen and Jan Bendtsen, *Evaluation of Aggregators for Integration of Large-scale Consumers in Smart Grid*, World Congress of The International Federation of Automatic Control 2014, pp. 1879–1885, 2014.

Contributions

Paper A

Exploring the Value of Flexibility: A Smart Grid Discussion

Mette Kirschmeyer Petersen, Lars Henrik Hansen and Tommy Mølbak

This paper was published in: Proceedings of IFAC Symposium on Power Plants and Power System Control, 2012.

The layout has been revised

Abstract

This paper investigates how the value of flexibility in energy systems is determined by markets, forecasts and physics. Both a traditional and a Smart Grid system are discussed.

In a traditional energy system, where flexibility is provided by power plants, the quality of a given flexible resource can be determined fairly unambiguously by activation time, length of reservation period and capacity. In a Smart Grid system, flexibility should be provided by flexible consumer appliances, which are not created for power management. Determining the quality and value of a flexible resource therefore becomes far more multifaceted, since additional performance constraints such as storage capacity, minimum runtime, temporal constraints (deadlines), ramp rates etc. must be considered.

To explore the value of flexibility, this paper defines two distinct operation strategies for a Virtual Power Plant: The *predictive* and the *agile*. A predictive Virtual Power Plant will generate predictions of future market prices, and based on these it will operate the portfolio in a least cost manner. An Agile Virtual Power Plant, on the other hand, operates its portfolio based on an analysis of the quality of the individual flexible resources. The Agile Virtual Power Plant is therefore reluctant to expend its flexible resource and does so in a worst-quality-resource manner.

1 Introduction

A successful Virtual Power Plant operator must answer the question: Which is more valuable: A heat pump, an electric vehicle or a freezer? The answer to this question lies somewhere between the Spot Market and Kirchoff's circuit law, so a good Virtual Power Plant-operator must be just the right combination of stockbroker and engineer. This paper explores how the value of flexibility is determined by markets, forecasts and physics both in a traditional energy system and a Smart Grid system.

Electricity is a just-in-time product, which means that it is instantly consumed at production. Electricity production and consumption must therefore be closely balanced at all times or the system will crash. Production technologies which harvest energy from natural sources such as wind, sun and waves are however unpredictable and fluctuating by nature. This means that the introduction of large ratios of renewable energy into the existing power system poses a significant challenge in terms of maintaining the real-time balance between production and consumption.

Traditionally, the Danish power system is dominated by power plants and inflexible consumption. The flexibility needed to balance production and consumption in real-time is therefore provided by power plants running at less than full capacity, the so-called reserves. The quality of reserves in such a power system is considered in [4]: "Various quality of reserves are graded by the quickness and sureness of response. This classification provides an unambiguous ordering by value with the best quality of reserve always preferred to the second best, and so on". In Smart Grid systems, the flexibility of consumer appliances, such as heat pumps, electric vehicles, cooling systems, micro CHP's, emergency generators etc. should be mobilized and play an active part in solving the balancing task. Discrepancies between supply and demand should then be evened out via (short-term) storage of energy, [1], or by voluntarily displacing consumption in time,

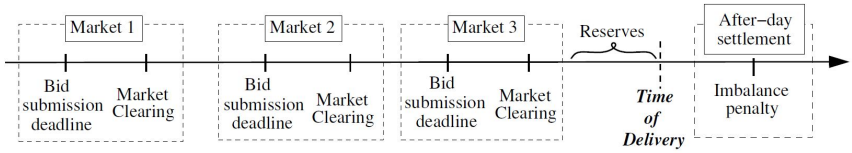

Figure 5.1: Energy markets operate in several cycles. As the time of delivery approaches, reserves take over the balancing task as energy markets cannot operate fast enough to balance production and consumption in real time.

so-called *demand-side management*, [2]. With this vision, however, the *unambiguous ordering by value* is lost.

Unlike traditional power plants, flexible appliances were not created for power management. This means that inherent comfort demands must be adhered to when utilizing the flexibility of the individual units. The quality of flexibility thus becomes multifaceted and ambiguous, when many factors besides quickness and sureness of response must be considered. Examples of such factors are storage capacity, minimum runtime, temporal constraints (deadlines), ramp rates etc.

In order to investigate the value of flexible resources in a Smart Grid system, this paper assumes that the link between flexible units and energy markets is facilitated by an independent, commercial aggregator. This commercial aggregator is denoted a Virtual Power Plant. The Virtual Power Plant gains its profit from trading a portfolio of flexible units in the energy markets. We make the following distinction between quality and value: Quality refers to the physical capabilities and limitations of a portfolio, which dictates how much flexibility is available for market trading. Value on the other hand is the monetary profit, which is gained by that trading.

This paper makes the assumption that a better quality portfolio will also generate more value. We therefore propose that a Virtual Power Plant must decide whether to adopt a *predictive* or an *agile* strategy. We also argue that a clear distinction between these two concepts is needed in order to adequately assess the relative quality of units in a heterogenous portfolio of flexible resources.

The contribution of this paper is to link the Agile Virtual Power Plant to value creation in a liberalized market set-up. The present paper thereby frames and motivates the theoretical work developed in [12]. In [12] the existence of an optimal dispatch strategy for the Agile Virtual Power Plant imbalance compensation problem is formally proved.

2 Balancing at the Market Level

At the market level, the balance between production and consumption is maintained by means of energy markets, after-day settlement, imbalance penalty, reserves and power markets, see Figure 5.1. This section gives a general description of these concepts with special attention on the price determining mechanisms.

Energy Markets

Energy markets (or forward markets) operate before the actual time of delivery to produce a schedule for how consumption and production should be balanced in the near future. Producers and wholesalers make bids for future time slots based on the best available knowledge, such as wind forecasts, consumption forecasts or general market knowledge and experience. Energy prices are then determined based on these bids, and a *balanced* plan for production and consumption is generated. Once prices on an energy market are settled (Market Clearing), then that market is closed. An important implication of this is that prices and quantities are fixed simultaneously.

Energy markets operate in several cycles, where early markets are followed-up by later ones. Producers and wholesalers make bids based on their best estimation of the future events and needs, and as time progresses better estimations become available. Early markets are therefore followed by later ones, where players have the option of adjusting their initial production and consumption schedules.

Electricity prices are predicted based on different data sets, such as weather forecast, commodity prices (oil, gas, biomass etc.) and 24-hour power consumption traces. According to [8], however, price spikes are highly randomized events, which can be caused by market power, transmission contingency, transmission congestion, generation contingencies, fuel prices, plant operating costs, weather conditions etc.

In [4] the link between expectations/forecast and price is formulated as: "In a well arbitraged market the forward price for delivery at time T will equal the expected spot price at time T". Since energy markets operate in cascades this statement could be extended to: In a well arbitraged market the forward price at time $T_{Present}$ for delivery at time T_{Future} will equal the spot price, which is expected for time T_{Future} at time $T_{Present}$.

This formulation reflects that if forecasts and assumptions made on earlier markets were accurate, then little correction is needed to the agreements made on the early market. This means that the price on later markets is the same as on earlier markets, since the expected spot price is unchanged. On the other hand, if forecasts on earlier markets were erroneous, then a lot of adjustment is needed. This means that prices on either up or down regulation in the later markets will be high.

After-Day Settlement and Imbalance Penalty

In the schedule generated by the energy markets, consumption and production are always balanced. After-day settlement and imbalance penalties assure that the actual execution of the plan is also balanced.

After the actual time of delivery, metered data of actual production/consumption is evaluated. In the after-day settlement producers and wholesalers are then invoiced according to their trades across all energy and power markets. Players are also given an imbalance penalty if they have deviated from the market agreements. The imbalance penalty is given if players deviate from the agreed schedule in either direction, so they are also fined for over-producing and under-consuming.

The actual size of the imbalance penalty is settled based on the size of the over/under production/consumption and the general electricity prices at the time when the imbalance was incurred.

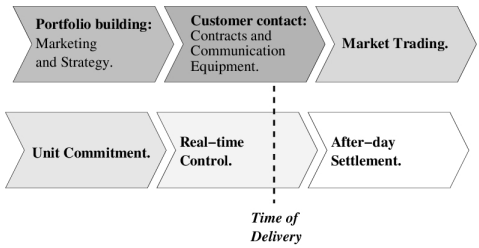

Figure 5.2: Timeline for operation phases of a Virtual Power Plant.

Reserves and Power Markets

The reason for implementing the imbalance penalty is that if you deviate from the market schedule you generate a need for up- or down regulation. Up regulation and down regulation are performed by spare capacity (flexibility), which is standing by in order to step in and stabilize the balance if needed. The spare capacity is denoted reserves. Reserves are in place because "the price mechanism cannot work fast enough to balance consumption and production in real time", [3]. Reserves are provided by specific power plants, which are operating at less than full capacity. This allows the power plant to ramp up or down as needed.

Operating at less than full capacity constitutes an expense to the asset owner, since spare capacity cannot be traded on the energy market. The asset owner is therefore given a reserve payment, independent of whether the reserve (up or down regulation) is activated or not. In a deregulated market, the reserve service is traded in designated power markets. This ensures that a competitive price is paid for reserves.

3 Virtual Power Plant Operation

This paper assumes that the link between flexible units and energy markets is facilitated by a commercial aggregator, denoted a Virtual Power Plant. The flexibility of the aggregated consumers is *the* value-adding resource of the Virtual Power Plant, since its only profit gain comes from trading the portfolio in the energy markets. This obviously means that a larger portfolio has a larger potential for revenue, but there is also a number of expenses, which grows with the size of the portfolio. Such expenses are marketing, installation and maintenance of communication and metering equipment plus customer billing and accounting. These fixed costs make it vital to the Virtual Power Plant that the aggregated flexibility is utilized to its absolute optimum. Another challenge for the Virtual Power Plant is that if the commercial Virtual Power Plant concept proves economically viable then several competing Virtual Power Plants will eventually emerge. With this vision, flexibility becomes a commodity in itself and multiple aggregators compete for flexible units in order to obtain the most profitable portfolio.

Operation phases for a commercial Virtual Power Plant are depicted in Figure 5.2 and similar considerations can be found in [9]. For the Virtual Power Plant value is created across operation phases, so the profitability of a given Virtual Power Plant will be determined by its ability to

- build and manage a lucrative portfolio of flexible units (Quality), and
- trade the portfolio in the power and energy markets (Value).

The Quality of Flexibility

In a traditional power system, flexibility is provided by power plants running at less than full capacity. The quality of such a reserve is determined by

- Capacity: How much imbalance will the resource be able to compensate, both in terms of power and energy,
- Length of reserve period: How long is the unit committed to stand-by, and
- Activation time: How fast can the reserve be activated.

For the commercial Virtual Power Plant, however, determining the quality of a flexible resource is far more complex. The primary purpose of flexible consumer appliances is not power management. Certain inherent comfort demands must therefore be adhered to. These comfort demands characterize the flexible resource, but also limit its quality and thus value. Characteristics, which must be considered by the Virtual Power Plant when assessing a flexible resource, are:

1. Power capacity
2. Length of reserve period
3. Activation time
4. Base load requirements
5. Temporal constraints (Deadlines)
6. Ramp rates
7. Storage capacity
8. Storage period/drain
9. Min/max run/down time
10. Warranty

In the following sections, it is argued that as a result of this setup the quality and value of flexibility in a Smart Grid system is poorly investigated via a *predictive* approach. The alternative of the *Agile* Virtual Power Plant is therefore suggested.

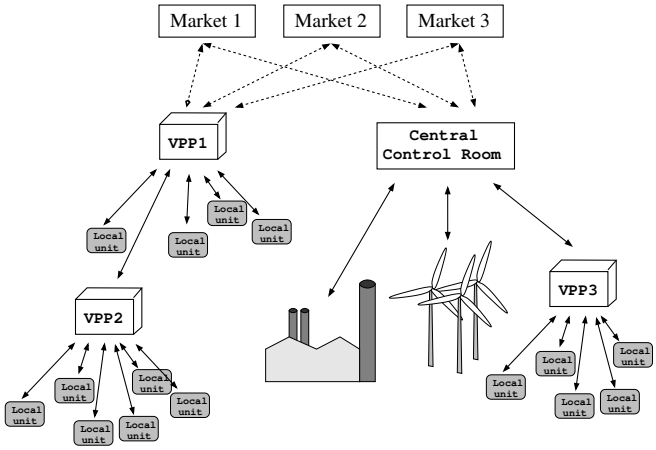

Figure 5.3: The market access of the Virtual Power Plant can be direct (VPP1), through integration with another Virtual Power Plant (VPP2) or through integration with a larger portfolio of production units (VPP3).

4 Predictive Virtual Power Plant

In recent years, the predictive Virtual Power Plant strategy has been explored in a variety of setups. The predictive strategy consists of predicting future market prices (or imbalances) as best as possible and operating the portfolio in a least-cost manner based on these predictions.

The operation of a portfolio of micro-combined heat and power plants is examined in [7], where the operation of the portfolio is optimized against spot market prices. This study uses spot prices from NordPool, [10], and perfect prediction of spot market prices is assumed.

A similar investigation is provided in [11], where optimal charging of electric vehicles is studied. Here electricity prices are estimated using multivariate regression, with regression variables for forecasted demand, forecasted wind power, hydro power, coal price, gas price and emission allowance price. Once forecasted these prices are assumed to be accurate for the following six hours.

5 Agile Virtual Power Plant

When correlating the predictive Virtual Power Plant strategy with the flexibility characteristics (1) - (10) it is clear that the predictive strategy is only optimal if predictions are correct and the length of prediction horizon is sufficiently large. The basic idea of the Agile Virtual Power Plant strategy is therefore to maximizes the quality of the portfolio rather than the predicted value. The Agile Virtual Power Plant thus operates the portfolio based on an analysis of the quality of the flexible resources themselves. The main motivation for exploring the Agile strategy is therefore to make the portfolio more robust (and thus profitable in later markets) when forecasts are erroneous or insufficiently long. A second motivation for exploring the Agile Virtual Power Plant strategy is that if the

Virtual Power Plant is to supply services similar to traditional power plant reserves, then any assumption of predictability is simply self-contradictory.

Clearly, the value that can be gained from the agile strategy is entirely dependent on the quality of forecasts and predictions. If forecasts and predictions are accurate then the cost minimization tactic of the predictive Virtual Power Plant is obviously the better of the two. It can, however, be argued that making accurate price projections for the Danish Smart Grid system will become increasingly difficult, due to

- Significant wind penetration level, [5], implies that forecasting errors will have a large impact on production schedules and thus impact prices in later markets closer to operational time,
- Integration of the European market and grid, [6], which means that market players, transmission congestion and events in larger areas will have to be considered in price projections, and
- Addition of flexible production and consumption increasing the price projection complexity in itself, since actions of major competitors influence market prices.

6 Simulation Example

This simulation example illustrates how the agile analysis can improve Virtual Power Plant performance. The considered portfolio is heterogeneous, but units differ only in power- and energy capacity. The market access of the Virtual Power Plant can be direct, through integration with another Virtual Power Plant or through integration with a larger portfolio of production units (see Figure 5.3). In either case, we let $P_{Reserve}(k)$ denote the aggregated power capacity, which can be offered for trading at sample k. In our simulation we do not assign market prices to samples. Instead we assume that the market trading is handled properly such that maximizing $P_{Reserve}$ (quality) also generates higher profit (value).

The Virtual Power Plant has control of a set of local units $\{LU_i\}_{i=1,2,\ldots,N}$, which are governed by individual dynamics and constraints. We model the local units simply as power and energy constrained integrators and let $E_i(k)$ denote the energy level in local unit i at sample k:

$$\begin{aligned} LU_i(k): \qquad & E_i(k+1) = E_i(k) + T_s P_i(k) \\ & \underline{P}_i \leq P_i(k) \leq \overline{P}_i \\ & \underline{E}_i \leq E_i(k+1) \leq \overline{E}_i \\ & E_i(0) = E_{i,0}, \end{aligned}$$

where $k = 0, 1, \ldots, \infty$, $i \in \mathbb{N}$, $\underline{P}_i \in \mathbb{R}^-$, $\overline{P}_i \in \mathbb{R}^+$, $\underline{E}_i \in \mathbb{R}^-$, $\overline{E}_i \in \mathbb{R}^+$ and $\underline{E}_i \leq E_{i,0} \leq \overline{E}_i$. With the choice of power and energy constrained units we have

$$P_{Reserve,i}(k) = \min\left(\overline{P}_i, \frac{\overline{E}_i - E_i(k)}{T_s}\right).$$

At each sample some volume, $P_{Dispatch}(k)$, is received, the size of which depends on market trading and current imbalances. We assume that $0 \leq P_{Dispatch}(k) \leq \sum_{i=1}^{N} P_{Reserve,i}(k)$.

The Virtual Power Plant must dispatch the full volume $P_{Dispatch}(k)$ to the local units, corresponding to $\sum_{i=1}^{N} P_i(k) = P_{Dispatch}(k)$. Since the goal of the Virtual Power Plant is to service the power system as well as possible it should maximize the flexibility, which can be offered to the market, so the optimization problem of the Virtual Power Plant is

$$\max_{P_i(\cdot)} \sum_{k=0}^{\infty} \sum_{i=1}^{N} P_{Reserve,i}(k) \tag{5.1}$$

$s.t.$

$$P_{Reserve,i}(k) = \min\left(\overline{P}_i, \frac{\overline{E}_i - E_i(k)}{T_s}\right) \tag{5.2}$$

$$0 \leq P_{Dispatch}(k) \leq \sum_{i=1}^{N} P_{Reserve,i}(k) \tag{5.3}$$

$$\sum_{i=1}^{N} P_i(k) = P_{Dispatch}(k) \tag{5.4}$$

$$E_i(k+1) = E_i(k) + T_s P_i(k) \tag{5.5}$$

$$\underline{P}_i \leq P_i(k) \leq \overline{P}_i \tag{5.6}$$

$$\underline{E}_i \leq E_i(k+1) \leq \overline{E}_i \tag{5.7}$$

$$E_i(0) = E_{i,0}. \tag{5.8}$$

Predictive Virtual Power Plant

The predictive Virtual Power Plant strategy consists of predicting future imbalances as best as possible and operating the portfolio according to these predictions. At each sample k, the Predictive dispatch strategy is therefore obtained by solving the optimization problem (5.1) - (5.8) with perfect prediction of $P_{Dispatch}$ over the horizon $N_{Predict}$:

$$\max_{P_i(\cdot)} \sum_{n=0}^{N_{Predict}} \sum_{i=1}^{N} P_{Reserve,i}(k+n)$$

$$s.t.$$

$$P_{Reserve,i}(k+n) = \min\left(\overline{P}_i, \frac{\overline{E}_i - E_i(k+n)}{T_s}\right)$$

$$0 \leq P_{Dispatch}(k+n) \leq \sum_{i=1}^{N} P_{Reserve,i}(k+n)$$

$$\sum_{i=1}^{N} P_i(k+n) = P_{Dispatch}(k+n)$$

$$E_i(k+1+n) = E_i(k+n) + T_s P_i(k+n)$$

$$\underline{P}_i \leq P_i(k+n) \leq \overline{P_i}$$

$$\underline{E}_i \leq E_i(k+1+n) \leq \overline{E_i}$$

$$E_i(0) = E_{i,0}.$$

Agile Virtual Power Plant

The Agile Virtual Power Plant strategy is to analyze the portfolio and dispense flexible resource in a least valuable unit manner. When we simply consider power and energy constrained integrators, then the value of the individual units is given by

$$\mathcal{K}_i(k) = \frac{\overline{E}_i - E_i(k)}{T_s \overline{P}_i},$$

which is the ratio between the energy reserve and the upper power limit. This factor states how many samples unit i can operate at its maximum before it becomes inactive (full). A higher $\mathcal{K}_i$-value thus means that the local unit can remain active through a longer period of high load. It is therefore reasonable to suggest the following linear cost function to obtain the Agile Virtual Power Plant dispatch strategy

$$\max_{P_i(k)} \sum_{i=1}^{N} \mathcal{K}_i(k) P_i(k), \tag{5.9}$$

as this will ensure that $P_{Dispatch}$ is distributed to the least valuable units. The Agile-Linear dispatch strategy is therefore obtained by minimizing (5.9) subject to (5.2) to (5.8). In [12] the authors demonstrated that while (5.9) generates a good dispatch strategy for the considered problem it is not optimal. The optimal dispatch strategy can however be

Table 5.1: Parameters for the local units.

$_i$	$\overline{P}_i$	$\underline{P}_i$	$\underline{E}_i$	$\overline{E}_i$	$\mathcal{K}_i(0)$
1	1	-1	0	1	1
2	1	-1	0	10	10
3	1	-1	0	100	100
4	10	-10	0	10	1
5	10	-10	0	100	10
6	10	-10	0	1.000	100
7	100	-100	0	100	1
8	100	-100	0	1000	10
9	100	-100	0	10.000	100

obtained by using the cost function

$$\max_{P_i(k)} \sum_{i=1}^{N} \frac{\left(\frac{\overline{E}_i - E_i(k)}{T_s} - P_i(k)\right)^2}{-2\overline{P}_i}, \tag{5.10}$$

so the Agile-Quadratic dispatch strategy is obtained by minimizing (5.10) subject to (5.2) to (5.8). In [12] the optimality of the Agile-Quadratic dispatch strategy is formally proved under the added assumption $\underline{P}_i = \underline{E}_i = 0$.

Simulation Results

The following simulation example illustrates the Agile-Quadratic, Agile-Linear and Predictive strategies. At each sample k, we choose $P_{Dispatch}(k)$ randomly from a uniform distribution subject to the constraint

$$\begin{aligned} P_{Dispatch}(k) \leq \min\big[&P_{Reserve,Agile-Quadratic}(k), \\ &P_{Reserve,Agile-Linear}(k), \\ &P_{Reserve,Predictive}(k)\big], \end{aligned}$$

which ensures that all optimization problems are feasible.

Nine local units are included in the simulations and parameters for these are given in Table 5.1. Additional simulation parameters are $N_{sim} = 30$, $T_s = 1$ and $N_{Predict} = 5$.

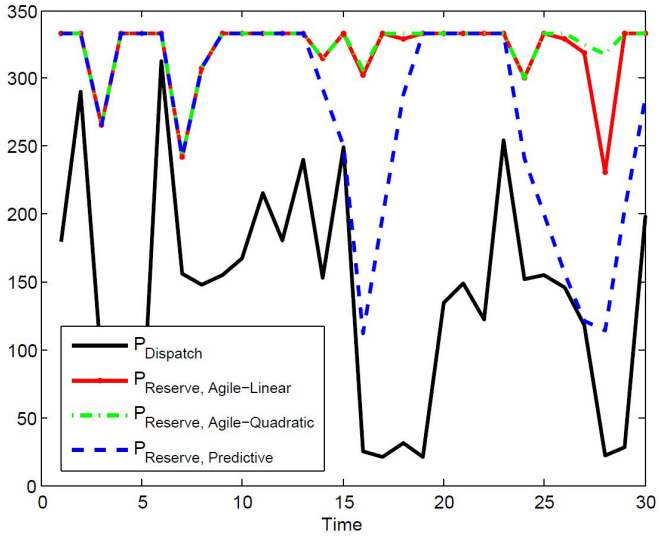

Figure 5.4: The upper bound on the amount of imbalance which can be dispatched to the Virtual Power Plant is denoted $P_{Reserve}$. A high value of $P_{Reserve}$ thus means a better utilization of the available flexibility. At each sample the predictive dispatch strategy has perfect prediction of $P_{Dispatch}$ over the next five samples; An assumption, which is *not* made by the two agile strategies.
In the first part of the simulations, the three methods perform equally well. After sample 13, however, the agile strategies are able to compensate a larger imbalance than the predictive strategy. This happens even though the three methods have the exact same portfolio at their disposal and have to balance the exact same load.

The simulation results are depicted in Figure 5.4. In the first part of the simulations, the three strategies perform equally well. After sample 13, however, the agile strategies can compensate larger imbalances, even though the three methods use the same portfolio to balance the same variations. The Agile-Linear strategy performs worse than the Agile-Quadratic, but still much better than the Predictive.

The explanation for why the agile strategy performs better than the predictive strategy can be found in Figure 5.5, which depicts the energy levels in each of the nine local units. The Agile-Quadratic strategy is able to get a better utilization of LU_9 early in the simulations and is better at rebuilding the flexibility of unit 1 and 4.

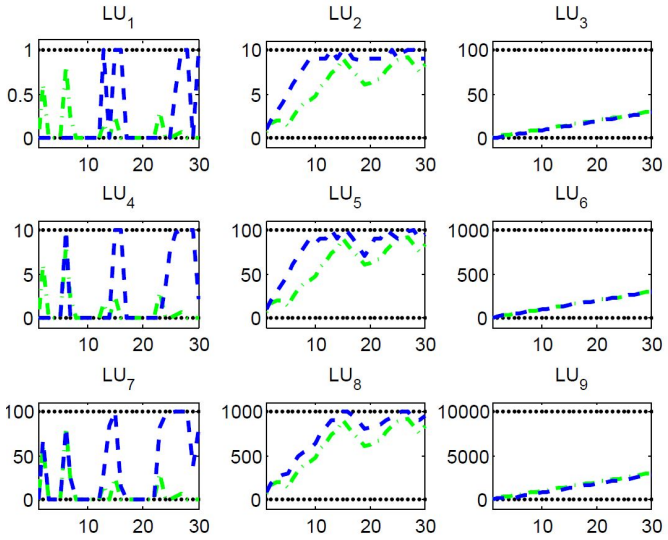

Figure 5.5: Energy level for each of the nine energy and power constrained local units (LUs) considered in the simulations (Parameters for the local units are given in Table 5.1). Blue depicts the Predictive strategy and green depicts the Agile-Quadratic strategy. Notice that the agile strategy has a better utilization of LU_9 (the largest individual unit) early in the simulations and does better at rebuilding the flexibility of unit 1 and 4.

7 The Real World Rarely Provides Extreme or Pure Cases [13]

This paper has presented the agile and predictive Virtual Power Plant strategies and made a strict distinction between the two. This distinction is useful for the sake of discussion, and because the predictive strategy is already well researched. In a practical setting, however, a combination of the two is far more meaningful. The intent of this article is therefore not to render the use of predictions impossible, but rather to comment that additional value can be gained by performing an analysis of the quality of the individual flexible resources and thus avoiding the prediction pitfall.

References

[1] Wade, N.S., Taylor, P.C., Lang, P.D. and Jones, P.R. (2010). Evaluating the benefits of an electrical energy storage system in a future smart grid, *Energy Policy, Volume 38*

[2] Mohsenian-Rad, A., Wong, V.W.S., Jatskevich, J., Schober, R. and Leon-Garcia, A. (2010). Autonomous Demand-side Management Based on Game-theoretic Energy

Consumption Scheduling for the Future Smart Grid, *IEEE Transactions on Smart Grid*, Volume 1, Issue 3

[3] Wangensteen, I. (2007). Power System Economics - the Nordic Electricity Market, *Tapir Academic Press*.

[4] Stoft, S. (2002). Power System Economics Designing Markets for Electricity, *Wiley-IEEE Press 1 (1)*.

[5] The Danish Government (2011), *Government Platform, Et Danmark, der staar sammen.*

[6] European Commission (2011) *Energy infrastructure - Priorities for 2020 and Beyond.*

[7] You, S., Traeholt, C. and Poulsen, B. (2009:1). A Study on Electricity Export Capability of the μCHP System with Spot Price, *IEEE Power & Energy Society General Meeting*.

[8] Zhao, J.H., Dong, Z.Y., Li, X. and Wong, K.P. (2005). A General Method for Electricity Market Price Spike Analysis, *IEEE Power Engineering Society General Meeting*, Volume 1.

[9] You, S., Traeholt, C. and Poulsen, B. (2009:2), Generic Virtual Power Plants: Management of Distributed Energy Resources under Liberalized Electricity Market, *The 8th IET International Conference on Advances in Power System Control, Operation and Management, Hong Kong, P.R. China*.

[10] NordPool `http://www.nordpoolspot.com/`

[11] Kristoffersen, T.K., Capion, K. and Meibom, P. (2010). Optimal charging of electric drive vehicles in a market environment, *Applied Energy*, Volume 88, Issue 5.

[12] Petersen, M., Bendtsen, J.D. and Stoustrup, J. (2012). Optimal Dispatch Strategy for the Agile Virtual Power Plant, *American Control Conference 2012, Montreal*.

[13] Teece, J.D. (1986), Profiting from technological innovation: Implications for integration, collaboration, licensing and public policy, *Elsevier Science Publischers B.V.*

Paper B

Optimal Dispatch Strategy for the Agile Virtual Power Plant

Mette Kirschmeyer Petersen, Jan Dimon Bendtsen and Jakob Stoustrup

This paper was published in: Proceedings of American Control Conference, 2012.

The layout has been revised

Abstract

The introduction of large ratios of renewable energy into the existing power system is complicated by the inherent variability of production technologies, which harvest energy from wind, sun and waves. Fluctuations of renewable power production can be predicted to some extent, but the assumption of perfect prediction is unrealistic. This paper therefore introduces the Agile Virtual Power Plant. The Agile Virtual Power Plant assumes that the base load production planning based on best available knowledge is already given, so imbalances cannot be predicted. Consequently the Agile Virtual Power Plant attempts to preserve maneuverability (stay agile) rather than optimize performance according to predictions.

In this paper the imbalance compensation problem for an Agile Virtual Power Plant is formulated. It is proved formally, that when local units are power and energy constrained integrators a dispatch strategy exists, which is optimal regardless of future load/imbalances. The optimal dispatch is obtained at each sample by solving a quadratic program. Finally a simulation example illustrates the optimal dispatch strategy and compares the performance with a (non-optimal) MPC-strategy.

1 Introduction

Electricity is a so-called just-in-time product, which means that it is instantly consumed at production. This means, that electricity production and consumption must be closely balanced at all times. Production technologies, which harvest energy from natural sources such as wind, sun and waves, are unpredictable and fluctuating by nature. This means that the introduction of large ratios of renewable energy into the existing power system poses a significant challenge in terms of maintaining the real-time balance between production and consumption.

In Smart Grid systems the flexibility of consumers, such as electric vehicles, heat pumps and HVAC-systems, should be mobilized and play an active part in solving the balancing task. With this vision the discrepancies between supply and demand should be evened out via (short-term) storage of energy [5] or by voluntarily displacing consumption in time, so-called *demand-side management* [6].

A Virtual Power Plant is a collection of flexible consumers, which are grouped together and controlled centrally, see [3]. In this paper the flexible consumers are denoted local units and are modeled simply as power and energy constrained integrators.

The considered Virtual Power Plant is part of a larger portfolio of production units, such as wind turbines, solar panels, power plants etc. The entire portfolio is managed by a master controller, which trades the production capacity on the energy markets, see Figure 6.1. Based on the market trading a production schedule for the portfolio is obtained (for more on energy markets see [7] and [8]).

Electricity production and consumption in the (near) future can be estimated based on weather forecast and 24-hour power consumption traces. However, the assumption of perfect prediction is unrealistic, so this paper explores a dispatch strategy for an Agile Virtual Power Plant. Because base load production is already given the master controller utilizes the Virtual Power Plant to compensate unforeseen errors and imbalances, such that the portfolio as a whole is following the agreed schedule, see Figure 6.2. The Agile Virtual Power Plant therefore attempts to ”stay agile”, rather than optimize performance

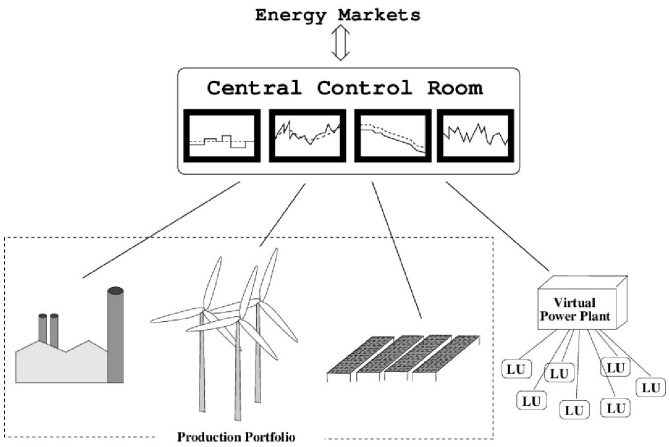

Figure 6.1: The Virtual Power Plant is part of a larger portfolio of production units, such as wind turbines, solar panels, power plants etc. The portfolio is controlled centrally by a master controller, which utilizes the capacity of the Virtual Power Plant to compensate imbalances in production, such that the portfolio as a whole is following an agreed schedule.

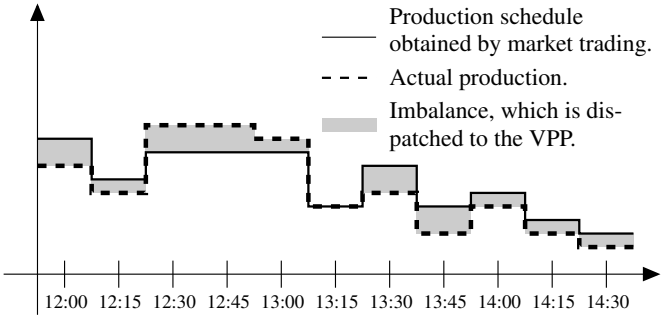

Figure 6.2: A production schedule for the production portfolio is obtained by market trading. Any imbalance is then dispatched to the Virtual Power Plant, such that the portfolio as a whole is following the production schedule.

according to predictions. The Agile Virtual Power Plant thus plays a role similar to that of traditional power plant reserves in a typical European power system (see [9]).

A similar balancing setup is considered in [1], which investigates energy storage in power system operations. In this work the flexible units are denoted power nodes. These power nodes are essentially energy and power constrained integrators, but ramp constraints and storage loss are also included in the modeling. A Model Predictive Control (MPC) approach is taken, with the assumption of perfect prediction of imbalances. Values for the dispatch parameters are obtained by manual tuning in order to obtain the desired system behavior.

Also [2] considers the operation of storage devices in power systems. In this paper

the storage units are also modeled as power and energy constrained integrators, but individual charging and decharging costs are include for each unit. Like [1] the paper [2] also takes an MPC approach to the balancing problem and imbalances are modeled as stochastic processes with diurnal components. It is not explained how values for the dispatch parameters are obtained, but it is indicated that they reflect the monetary costs of charging/decharging and the cost of load shedding.

In the work described above values for the dispatch parameters are fixed based on heuristics or monetary costs. This paper proposes that the dispatch parameters be chosen based on the individual local units' ability to compensate imbalances. Dispatch parameters are therefore calculated based on the state and characteristics (constraints) of each local unit itself.

The contribution of this paper is to formulate the imbalance compensation problem for an Agile Virtual Power Plant. It is proved formally that when the local units have a specific, simple form an optimal dispatch strategy can be obtained at each sample by solving a quadratic optimization problem. Finally, simulation studies show, that for the considered optimization problem the assumption of perfect prediction over a certain horizon does not guarantee optimality.

The remainder of this paper is structured as follows: In Section 2, we formulate the imbalance compensation problem for an Agile Virtual Power Plant. Section 3 presents the main contribution of this paper, namely an optimal dispatch strategy for the imbalance compensation problem. In Section 4, a simulation example illustrates the optimal dispatch strategy and compares the performance with a (non-optimal) MPC-strategy. Finally, Section 5 gives concluding remarks and suggestions for further work.

2 Problem Formulation

General Form of the Agile Virtual Power Plant Imbalance Compensation Problem

As explained earlier we consider a Virtual Power Plant, which is part of a larger portfolio of production units. A master controller has direct control of the entire portfolio and trades the production capacity on the energy markets. Based on the market trading, a base load schedule for the production units is obtained. The Virtual Power Plant is then utilized to compensate for unforeseen errors and imbalances in production, such that the portfolio as a whole is following the agreed schedule.

The Virtual Power Plant has control of a set of local units $\{LU_i\}_{i=1,2,\ldots,N}$, which are governed by individual dynamics and constraints. The Virtual Power Plant offers the capacity of the local units to the master controller and we let $P_{Reserve,i}(k)$ denote the capacity of local unit i, which can be offered to the master controller at sample k. The Virtual Power Plant must offer its full available capacity to the master controller, so the offered capacity at sample k is

$$P_{Reserve}(k) = \sum_{i=1}^{N} P_{Reserve,i}(k).$$

At each sample some volume, $P_{Dispatch}(k)$, is received from the master controller. It is assumed that $0 \leq P_{Dispatch}(k) \leq P_{Reserve}(k)$, such that it is always possible

to dispatch $P_{Dispatch}$ to the portfolio. The Virtual Power Plant must dispatch the full volume $P_{Dispatch}(k)$ to the local units and we let $P_i(k)$ denote the quantity dispatched to unit i, so

$$\sum_{i=1}^{N} P_i(k) = P_{Dispatch}(k).$$

The goal of the Virtual Power Plant is to service the master controller as well as possible. The objective is therefore to dispatch $P_{Dispatch}(k)$ to the local units such that $P_{Reserve}(k), k = 0, 1, \ldots, K$, is maximized. This can be formulated as

$$\begin{aligned} &\max_{P_i(\cdot)} \sum_{k=0}^{K} \sum_{i=1}^{N} P_{Reserve,i}(k) \\ &s.t. \\ &\quad 0 \leq P_{Dispatch}(k) \leq \sum_{i=1}^{N} P_{Reserve,i}(k) \\ &\quad \sum_{i=1}^{N} P_i(k) = P_{Dispatch}(k) \end{aligned}$$

and also subject to the dynamics and constraints of $\{LU_i\}_{i=1,2,\ldots,N}$. This is the general form of the Agile Virtual Power Plant imbalance compensation problem.

Remark 2: (Optimization Target)
In the formulation given above only the upper bound on the available capacity is considered as an optimization target. With this setup we obtain a clear objective, namely maximizing $P_{Reserve}$. If both the upper and lower bounds on the available capacity are included, that is introducing both $\overline{P}_{Reserve}$ and $\underline{P}_{Reserve}$, then the objective becomes less clear. This is because a gain in $\overline{P}_{Reserve}$ will introduce an equivalent loss in $\underline{P}_{Reserve}$. The problem could be handled by considering a less intuitive objective function than the one presented above, but the penalty for the trade off between positive and negative reserve will invariably be based on heuristics. For now we will therefore only consider the upper bound on the available reserve and as a result it is assumed that the imbalance, $P_{Dispatch}$, is also positive, though this is obviously not a realistic assumption.

Agile Virtual Power Plant Imbalance Compensation Problem for Power and Energy Constrained Local Units

In this paper the local units are modelled simply as power and energy constrained integrators and we let $E_i(k)$ denote the energy level in local unit i at sample k.

Definition 10 (Power and Energy Constrained Local Unit)**.** The dynamics and constraints

of a power and energy constrained local unit are

$$\begin{aligned} LU_i(k): \qquad & E_i(k+1) = E_i(k) + T_s P_i(k) \\ & 0 \leq P_i(k) \leq \overline{P}_i \\ & 0 \leq E_i(k+1) \leq \overline{E}_i \\ & E_i(0) = E_{i,0}, \end{aligned}$$

where $k = 0, 1, \ldots, K$, $i \in \mathbb{N}$, $0 \leq \overline{P}_i$, $0 \leq \overline{E}_i$ and $0 \leq E_{i,0} \leq \overline{E}_i$.

For ease of notation we assume that $T_s = 1$ in the following and let $\boldsymbol{LU_N}(k)$ denote a set of $N \in \mathbb{N}$ local units, that is $\{LU_i(k)\}_{i=1,2,\ldots,N}$.

With the choice of power and energy constrained local units we obtain that

$$P_{Reserve,i}(k) = \min(\overline{P}_i, \overline{E}_i - E_i(k)),$$

so the Agile Virtual Power Plant imbalance compensation problem for power and energy constrained local units is

$$\max_{P_i(\cdot)} \sum_{k=0}^{K} \sum_{i=1}^{N} P_{Reserve,i}(k) \tag{6.1}$$

$s.t.$

$$P_{Reserve,i}(k) = \min(\overline{P}_i, \overline{E}_i - E_i(k)) \tag{6.2}$$

$$0 \leq P_{Dispatch}(k) \leq \sum_{i=1}^{N} P_{Reserve,i}(k) \tag{6.3}$$

$$\sum_{i=1}^{N} P_i(k) = P_{Dispatch}(k) \tag{6.4}$$

$$E_i(k+1) = E_i(k) + P_i(k) \tag{6.5}$$

$$0 \leq P_i(k) \leq \overline{P_i} \tag{6.6}$$

$$0 \leq E_i(k+1) \leq \overline{E_i} \tag{6.7}$$

$$E_i(0) = E_{i,0}, \tag{6.8}$$

where $E_{i,0}$ is the initial energy level of unit i.

To simplify the setup it has been assumed that the Virtual Power Plant is offering $P_{Reserve}$ to the master controller at every sample. When power and energy constrained units are considered, however, $P_{Reserve}$ could be offered for more than one sample without any loss of information using Resource Polytopes as described in [4]. This would give the master controller the benefit of knowledge of future balancing capacity.

3 Optimal Dispatch Strategy

This section presents the main contribution of the article, namely the result, that the optimal dispatch at each sample is independent of future load/imbalances; And the optimal dispatch can be obtained at each sample by solving a quadratic program.

Definition 11 (Agility Factor)**.**
Let $LU_i(k)$ denote a power and energy constrained local unit. The *Agility Factor* of local unit i at sample k is defined as

$$\mathcal{K}_i(k) = \frac{\overline{E}_i - E_i(k)}{\overline{P}_i}.$$

Lemma 2. *At sample k let $\boldsymbol{LU_N}(k)$ denote a finite set of power and energy constrained local units. A dispatch strategy for problem (6.1) - (6.8) can be obtained by solving the program*

$$\max_{P_i(k)} \sum_{i=1}^{N} \frac{(\overline{E}_i - E_i(k) - P_i(k))^2}{-2\overline{P}_i} \tag{6.9}$$

$s.t.$

$$P_{Reserve,i}(k) = \min(\overline{P}_i, \overline{E}_i - E_i(k)) \tag{6.10}$$

$$0 \leq P_{Dispatch}(k) \leq \sum_{i=1}^{N} P_{Reserve,i}(k) \tag{6.11}$$

$$\sum_{i=1}^{N} P_i(k) = P_{Dispatch}(k) \tag{6.12}$$

$$E_i(k+1) = E_i(k) + P_i(k) \tag{6.13}$$

$$0 \leq P_i(k) \leq \overline{P_i} \tag{6.14}$$

$$0 \leq E_i(k+1) \leq \overline{E_i} \tag{6.15}$$

$$E_i(0) = E_{i,0}. \tag{6.16}$$

and for this dispatch strategy the marginal cost/gain of dispatching to local unit i is $\mathcal{K}_i(k+1)$.

Proof. First observe that the constraints (6.2) - (6.8) are the same as (6.10) - (6.16), so at sample k a feasible dispatch strategy for problem (6.1) - (6.8) can be obtained by solving (6.9) - (6.16).

Next define

$$f(P_i(k)) = \sum_{i=1}^{N} \frac{(\overline{E}_i - E_i(k) - P_i(k))^2}{-2\overline{P}_i},$$

so

$$\begin{aligned}\nabla f(P_i(k)) &= \Bigg[\frac{\overline{E}_1 - E_1(k) - P_1(k)}{\overline{P}_1}, ..., \\ &\qquad \frac{\overline{E} - E_N(k) - P_N(k)}{\overline{P}_N}\Bigg] \\ &= \left[\frac{\overline{E}_1 - E_1(k+1)}{\overline{P}_1}, ..., \frac{\overline{E} - E_N(k+1)}{\overline{P}_N}\right] \\ &= [\mathcal{K}_1(k+1), ..., \mathcal{K}_N(k+1)].\end{aligned}$$

□

Definition 12 (Feasible Dispatch Sequence)**.**
Let $\boldsymbol{LU_N}(k)$ denote a finite set of power and energy constrained local units. The sequence $\{\boldsymbol{P_{Dispatch}(k)}\}_{k=0,1,...,K}$ is a *Feasible Dispatch Sequence* associated with $\boldsymbol{LU_N}(k)$ if problem (6.1) - (6.8) is feasible for $P_{Dispatch}(k) = \boldsymbol{P_{Dispatch}(k)}, k = 0, 1, \ldots, K$.

Definition 13 (Set of Feasible Dispatch Sequences)**.**
Let $\boldsymbol{LU_N}(k)$ denote a finite set of power and energy constrained local units. The *Set of Feasible Dispatch Sequences* over horizon K for $\boldsymbol{LU_N}(k)$ is denoted $\boldsymbol{\Omega_K}\big(\boldsymbol{LU_N}(k)\big)$.

Definition 14 (Integer Agility Factor System)**.**
A set of $\mathsf{K_{Max}}$ power and energy constrained local units, for which

$$\overline{\mathsf{E}}_1 = \overline{\mathsf{P}}_1, \overline{\mathsf{E}}_2 = 2 \cdot \overline{\mathsf{P}}_2, \ldots, \overline{\mathsf{E}}_{\mathsf{K_{Max}}} = \mathsf{K_{Max}} \cdot \overline{\mathsf{P}}_{\mathsf{K_{Max}}}$$

is denoted an *Integer Agility Factor System*. Observe that for an *Integer Agility Factor System* the index number, i, equals $\mathcal{K}_i$.

The set $\{\mathsf{LU}_j(k)\}_{j=1,2,...,\mathsf{K_{Max}}}$ is denoted $\mathbf{LU_{K_{Max}}}(k)$.

Lemma 3. *For any finite set of power and energy constrained local units,* $\boldsymbol{LU_N}(k)$*, there exists an integer* $\mathsf{K_{Max}}$ *and an* Integer Agility Factor System*, denoted* $\mathbf{LU_{K_{Max}}}(k)$*, such that*

$$\boldsymbol{\Omega_K}\big(\boldsymbol{LU_N}(k)\big) = \boldsymbol{\Omega_K}\big(\mathbf{LU_{K_{Max}}}(k)\big).$$

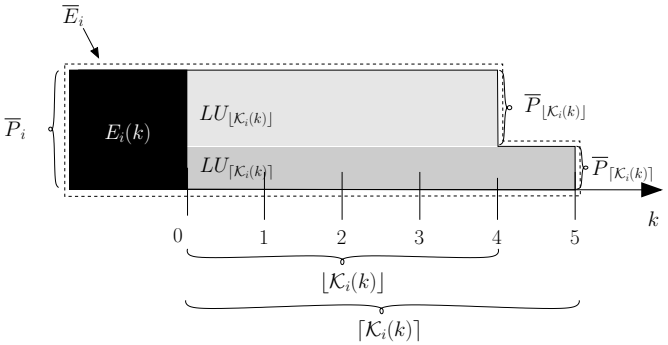

Figure 6.3: Lemma 3: Any power and energy constrained local unit can be expressed as two equivalent units, which have integer *Agility Factors*.

Proof. For each local unit in $\boldsymbol{LU_N}(k)$ define $LU_{\lceil \mathcal{K}_i(k) \rceil}$ by

$$\begin{aligned}\overline{P}_{\lceil \mathcal{K}_i(k) \rceil} &= \overline{E}_i - E_i(k) - \lfloor \mathcal{K}_i(k) \rfloor \cdot \overline{P}_i, \\ \overline{E}_{\lceil \mathcal{K}_i(k) \rceil} &= \left(\overline{E}_i - E_i(k) - \lfloor \mathcal{K}_i(k) \rfloor \cdot \overline{P}_i \right) \cdot \lceil \mathcal{K}_i(k) \rceil\end{aligned}$$

and $LU_{\lfloor \mathcal{K}_i(k) \rfloor}$ by

$$\begin{aligned}\overline{P}_{\lfloor \mathcal{K}_i(k) \rfloor} &= \overline{P}_i - \overline{P}_{\lceil \mathcal{K}_i(k) \rceil}, \\ \overline{E}_{\lfloor \mathcal{K}_i(k) \rfloor} &= \left(\overline{P}_i - \overline{P}_{\lceil \mathcal{K}_i(k) \rceil} \right) \cdot \lfloor \mathcal{K}_i(k) \rfloor,\end{aligned}$$

so $\boldsymbol{\Omega_K}\big(LU_i(k)\big) = \boldsymbol{\Omega_K}\big(\{LU_{\lceil \mathcal{K}_i(k) \rceil}, LU_{\lfloor \mathcal{K}_i(k) \rfloor}\}\big)$, see Figure 6.3. Next group these units according to equal $\mathcal{K}$-value to obtain $\mathbf{LU}_{\mathbf{K_{Max}}}(k)$, where $\mathsf{K_{Max}} = \max_{i=1,2,\ldots,N} \lceil \mathcal{K}_i(k) \rceil$. □

Lemma 4. *Let there be given an* Integer Agility Factor System, $\mathbf{LU}_{\mathbf{K_{Max}}}(k)$*, and a sequence* $\{\boldsymbol{P_{Dispatch}(k)}\}_{k=0,1,\ldots,K}$ *and define*

$$\ell_n = \left\{ k = 0, 1, \ldots, K | \boldsymbol{P_{Dispatch}(k)} > \sum_{j=n}^{\mathsf{K_{Max}}} \overline{\mathsf{P}}_j \right\}$$

for $n = 1, 2, \ldots, \mathsf{K_{Max}}$*. Then*

$$\{\boldsymbol{P_{Dispatch}(k)}\}_{k=0,1,\ldots,K} \in \boldsymbol{\Omega_K}\big(\mathbf{LU}_{\mathbf{K_{Max}}}(k)\big),$$

if and only if

$$P_{Dispatch}(k) \geq 0, \ k = 0, 1, \ldots, K \tag{6.17}$$

$$\sum_{k=0}^{K} P_{Dispatch}(k) \leq \sum_{j=1}^{\mathsf{K_{Max}}} \overline{\mathsf{E}}_j \tag{6.18}$$

$$\ell_1 = \emptyset \tag{6.19}$$

and

$$\sum_{k\in\ell_n}\left(\boldsymbol{P_{Dispatch}(k)}-\sum_{j=n}^{\mathsf{K_{Max}}}\overline{\mathsf{P}}_j\right)\leq\sum_{j=1}^{n-1}\overline{\mathsf{E}}_j \tag{6.20}$$

for $n=2,3,\ldots,\mathsf{K_{Max}}$*, see Figure 6.4.*

Proof. Consider an *Integer Agility Factor System* consisting of just one local unit $\{\mathsf{LU}_1(k)\}$, where

$$\overline{\mathsf{E}}_1=\overline{\mathsf{P}}_1.$$

Then

$$\{\boldsymbol{P_{Dispatch}(k)}\}_{k=0,1,\ldots,K}\in\boldsymbol{\Omega_K}\big(\{\mathsf{LU}_1(k)\}\big),$$

if and only if

$$\begin{aligned}&P_{Dispatch}(k)\geq 0,\ k=0,1,\ldots,K\\&\sum_{k=0}^{K}P_{Dispatch}(k)\leq\overline{\mathsf{E}}_1\end{aligned}$$

and

$$\ell_1=\emptyset.$$

Next consider an *Integer Agility Factor System* consisting of two local units $\{\mathsf{LU}_1(k),\mathsf{LU}_2(k)\}$, where

$$\overline{\mathsf{E}}_1=\overline{\mathsf{P}}_1,\overline{\mathsf{E}}_2=2\cdot\overline{\mathsf{P}}_2.$$

Then

$$\{\boldsymbol{P_{Dispatch}(k)}\}_{k=0,1,\ldots,K}\in\boldsymbol{\Omega_K}\big(\{\mathsf{LU}_1(k),\mathsf{LU}_2(k)\}\big),$$

if and only if

$$\begin{aligned}&P_{Dispatch}(k)\geq 0,\ k=0,1,\ldots,K\\&\sum_{k=0}^{K}P_{Dispatch}(k)\leq\sum_{j=1}^{2}\overline{\mathsf{E}}_j\\&\ell_1=\emptyset.\end{aligned}$$

and

$$\sum_{k\in\ell_2}\left(\boldsymbol{P_{Dispatch}(k)}-\overline{\mathsf{P}}_2\right)\leq\overline{\mathsf{E}}_1$$

Using equivalent reasoning, in the general case, that is when considering $\mathbf{LU}_{\mathbf{K_{Max}}}(k)$, we obtain (6.17) to (6.20). □

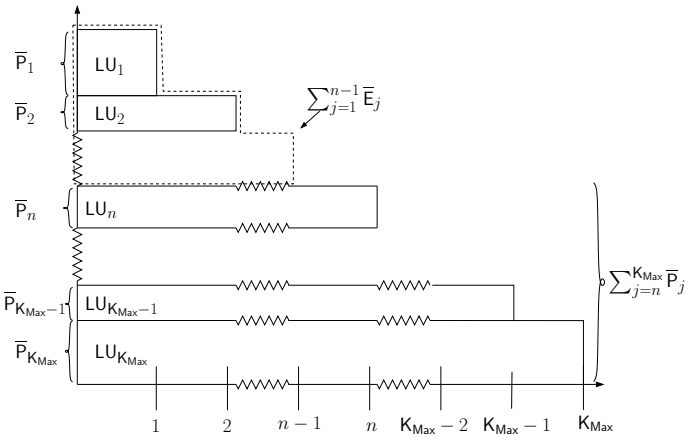

Figure 6.4: Lemma 4: Samples where $\boldsymbol{P_{Dispatch}(k)}$ exceeds $\sum_{j=n}^{\mathsf{K}_{\mathsf{Max}}}\overline{\mathsf{P}}_j$ are denoted ℓ_n. For these samples any quantity larger than $\sum_{j=n}^{\mathsf{K}_{\mathsf{Max}}}\overline{\mathsf{P}}_j$ must be dispatched to units $\{\mathsf{LU}_j(k)\}_{j=1,2,\ldots,n-1}$.

Lemma 5. *Let $\boldsymbol{LU_N}(k)$ denote a finite set of power and energy constrained local units and let some quantity $P_{Dispatch,0}$ satisfying $0 \leq P_{Dispatch,0} \leq \sum_{i=1}^{N} P_{Reserve,i}(k)$ be given.*

Also let $\boldsymbol{LU_N^R}(k+1)$ be a set of local units obtained by a feasible dispatch of $P_{Dispatch,0}$ to $\boldsymbol{LU_N}(k)$ and let $\boldsymbol{LU_N^Q}(k+1)$ be the set of local units obtained by dispatching according to the solution of problem (6.9) - (6.16). Then

$$\boldsymbol{\Omega_K}\big(\boldsymbol{LU_N^R}(k+1)\big) \subseteq \boldsymbol{\Omega_K}\big(\boldsymbol{LU_N^Q}(k+1)\big).$$

Proof. First let $\mathbf{LU}_{\mathbf{K}_{\mathbf{Max}}}(k)$ be the *Integer Agility Factor System* associated with $\boldsymbol{LU_N}(k)$ as given by Lemma 3. Then by Lemma 4 the set of feasible dispatch sequences is given by (6.17) to (6.20).

Next for each $n = 2, 3, \ldots, \mathsf{K}_{\mathsf{Max}}$ let α_n denote the ratio of $P_{Dispatch,0}$, which is dispatched to units $j = 1, 2, \ldots, n-1$. After dispatch of $P_{Dispatch,0}$ we have that $\{\boldsymbol{P_{Dispatch}(k)}\}_{k=0,1,\ldots,K}$ is a feasible dispatch sequence of the system if and only if

$$\begin{aligned}
&P_{Dispatch}(k) \geq 0,\ k = 0, 1, \ldots, K \\
&\sum_{k=0}^{K} P_{Dispatch}(k) \leq \sum_{j=1}^{\mathsf{K}_{\mathsf{Max}}} \overline{\mathsf{E}}_j - P_{Dispatch,0} \\
&\ell_1 = \emptyset
\end{aligned}$$

and

$$\sum_{k\in\ell_n}\Big(\boldsymbol{P_{Dispatch}(k)}-\sum_{j=n}^{\mathsf{K_{Max}}}\overline{\mathsf{P}}_j\Big) \leq \sum_{j=1}^{n-1}\overline{\mathsf{E}}_j-\alpha_n\cdot P_{Dispatch,0} \tag{6.21}$$

for $n=2,3,\ldots,\mathsf{K_{Max}}$.

It follows from (6.21), that the maximum set of feasible dispatch sequences after dispatch of $P_{Dispatch,0}$ is obtained by minimizing α_n for all n, that is

$$\min_{\mathsf{P}_j(k)}\alpha_n,\quad n=2,3,\ldots,\mathsf{K_{Max}},$$

subject to (6.12) to (6.16). This also means for each n dispatch as much as possible to the local units $n+1,n+2,\ldots,\mathsf{K_{Max}}$ and it follows by Lemma 2, that this is exactly what is obtained by the dispatch strategy (6.9) to (6.16).

Finally $\boldsymbol{\Omega_K}\big(\boldsymbol{LU_N^R}(k+1)\big)\subseteq\boldsymbol{\Omega_K}\big(\boldsymbol{LU_N^Q}(k+1)\big)$, since no other dispatch can generate higher upper bounds on (6.21) than what is obtained by (6.9) to (6.16). □

Theorem 6. *Dispatching according to the solution of (6.9) to (6.16) at each sample, yields an optimal dispatch strategy for (6.1) to (6.8).*

Proof. Let $\boldsymbol{LU_N}(k)$ denote a finite set of power and energy constrained local units. Observe, that at any sample $n\geq k$

$$P_{Reserve}(n)=\max_{\boldsymbol{P_{Dispatch}}(\cdot)\in\boldsymbol{\Omega_K}\left(\boldsymbol{LU_N}(n)\right)}\boldsymbol{P_{Dispatch}(n)}. \tag{6.22}$$

Next let $\{\boldsymbol{P_{Dispatch}(k)}\}_{k=0,1,\ldots,K}$ be any sequence in $\boldsymbol{\Omega_K}\big(\boldsymbol{LU_N}(k)\big)$. Also let $\{\boldsymbol{LU_N^{Opt}}(k)\}_{k=0,1,\ldots,K}$ denote the optimal sequence of sets of local units, that is the sequence of sets of local units obtained by dispatching according to the solution of (6.1) to (6.8). Finally let $\{\boldsymbol{LU_N^Q}(k)\}_{k=0,1,\ldots,K}$ denote the sequence of sets of local units obtained by dispatching according to the solution of (6.9) to (6.16) at each sample. By Lemma 5

$$\boldsymbol{\Omega_K}\big(\boldsymbol{LU_N^{Opt}}(k)\big)\subseteq\boldsymbol{\Omega_K}\big(\boldsymbol{LU_N^Q}(k)\big),\quad k=0,1,\ldots,K.$$

It now follows from (6.22) that dispatching according to the solution of (6.9) to (6.16) at each sample, yields an optimal dispatch strategy for (6.1) to (6.8). □

Remark 3: (Agile, Linear Dispatch Strategy)
By using the Agility Factors *of the local units* before *dispatch as weights in the objective function, we can formulate a linear problem, which also generates a "$\mathcal{K}$-greedy" dispatch strategy. This strategy is denoted the linear strategy and at sample k it is obtained by*

$$\max_{P_i(k)} \sum_{i=1}^{N} \mathcal{K}_i(k) P_i(k) \tag{6.23}$$

subject to (6.2) - (6.8).

The linear strategy, however, is not optimal, which is illustrated by the following example: Consider a system of the two local units given in Table 6.1 and let $P_{Dispatch}(k) = 10$. The solution of problem (6.23) subject to (6.2) - (6.8) is then $P_1(k) = 10$ and $P_2(k) = 0$. This means that $P_{Reserve}(k+1) = 15$, since $E_1(k+1) = 15$ and $E_2(k+1) = 6$. A higher value of $P_{Reserve}(k+1)$, however, is obtained by setting $P_1(k) = 5.5$ and $P_2(k) = 4.5$, since this makes $E_1(k+1) = 10.5$ and $E_2(k+1) = 10.5$, so $P_{Reserve}(k+1) = 19$. This shows that the linear strategy is not optimal.

The problem is that the linear strategy distributes according to $\mathcal{K}_i(k)$, that is, the state before dispatch, and does not consider the dynamic effects of the current dispatch. Since the quadratic optimization problem has marginal costs/gain of $\mathcal{K}_i(k+1)$ it exactly considers the dynamic effects of the current dispatch.

	LU_1	LU_2
$\overline{P}$	10	10
$\overline{E}$	20	20
$E(k)$	5	6
$\mathcal{K}_i(k)$	1.5	1.4

Table 6.1: System of local units for which the agile, linear strategy is not optimal for $P_{Dispatch}(k) = 10$.

4 Simulation Example

To illustrate the different dispatch strategies a simulation example has been constructed and implemented. The performance of the optimal and linear dispatch strategies are compared to a predictive dispatch strategy in which perfect prediction of $P_{Dispatch}$ is assumed over a certain prediction horizon.

At each sample k the optimal dispatch strategy solves the optimization problem (6.9) - (6.16) and the linear strategy solves problem (6.23) subject to (6.2) - (6.8). At each sample k the predictive dispatch strategy solves the optimization problem (6.1) - (6.8) by

assuming perfect prediction of $P_{Dispatch}$ over the horizon $N_{Predict}$:

$$\max_{P_i(\cdot)} \sum_{n=0}^{N_{Predict}} \sum_{i=1}^{N} P_{Reserve,i}(k+n) \tag{6.24}$$

$s.t.$

$$P_{Reserve,i}(k+n) = \min(\overline{P}_i, \overline{E}_i - E_i(k+n)) \tag{6.25}$$

$$0 \leq P_{Dispatch}(k+n) \leq \sum_{i=1}^{N} P_{Reserve,i}(k+n) \tag{6.26}$$

$$\sum_{i=1}^{N} P_i(k+n) = P_{Dispatch}(k+n) \tag{6.27}$$

$$E_i(k+1+n) = E_i(k+n) + P_i(k+n) \tag{6.28}$$

$$-\overline{P}_i \leq P_i(k+n) \leq \overline{P_i} \tag{6.29}$$

$$0 \leq E_i(k+1+n) \leq \overline{E_i} \tag{6.30}$$

$$E_i(0) = E_{i,0}, \tag{6.31}$$

where $N_{Predict}$ is less than K.

At each sample k in the simulations we choose $P_{Dispatch}(k)$ randomly from a uniform distribution subject to the constraint

$$P_{Dispatch}(k) \leq \min\big[P_{Reserve,Optimal}(k), P_{Reserve,Linear}(k), P_{Reserve,Predictive}(k)\big],$$

which insures that problem (6.9) - (6.16), problem (6.23) subject to (6.2) - (6.8) and problem (6.24) - (6.31) are all feasible.

Nine local units are included in the simulations and parameters for these are given in Table 6.2. Additional simulation parameters are $K = 90$, $T_s = 1$, and $N_{Predict} = 3$.

i	$\overline{P}_i$	$\overline{E}_i$	$E_{i,0}$	$\mathcal{K}_i(0)$
1	1	40	0	40
2	2	50	0	25
3	3	45	0	15
4	4	120	0	30
5	5	175	0	35
6	6	270	0	45
7	7	35	0	5
8	8	160	0	20
9	9	90	0	10

Table 6.2: Parameters for the local units.

The simulation results are depicted in Figure 6.5. In the first part of the simulations the three methods perform equally well. After sample 15, however, the optimal and linear

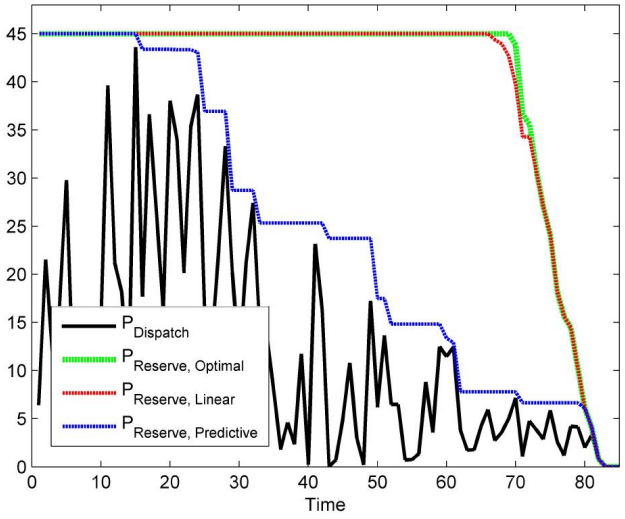

Figure 6.5: Simulation Results: $P_{Reserve}$ is an upper bound on the amount of imbalance which could potentially be dispatched to the Virtual Power Plant. A high value of $P_{Reserve}$ thus means a better utilization of the available flexibility. At each sample the predictive dispatch strategy assumes perfect prediction of $P_{Dispatch}$ over the next three samples; An assumption which is *not* made by the optimal and linear strategies.
In the first part of the simulations the three methods perform equally well. After sample 15, however, the optimal and linear dispatch strategies are able to compensate for a larger imbalance than the predictive dispatch strategy. This happens even though the three methods have the exact same local units at their disposal and have to balance the exact same load.

strategies are able to compensate for a larger imbalance than the predictive strategy, even though the three methods have the exact same local units at their disposal and have to balance the exact same load. As expected, the linear non-predictive strategy performs worse than the optimal strategy, but still much better than the predictive strategy.

The explanation for why the optimal strategy outperforms the predictive strategy can be found in Figure 6.6, which depicts the energy levels in each of the nine local units. The optimal strategy is able to get a better utilization of e.g. LU_7 and LU_9 early in the simulations, which allows it to stay clear of $\overline{E}$ for all local units until sample 70.

5 Discussion

This paper presented the imbalance compensation problem for an Agile Virtual Power Plant and proved the optimality of an associated dispatch strategy when the local units are power and energy constrained integrators. Furthermore, simulation results indicated, that for the considered optimization problem the assumption of perfect prediction is not enough to insure optimality.

Further research will address an extension of the setup by adding availability and

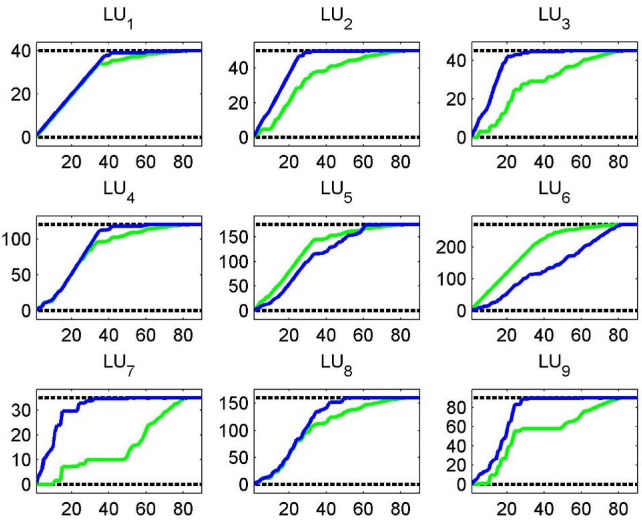

Figure 6.6: Simulation Results: Energy levels for each of the nine energy and power constrained local units (LUs) considered in the simulations (Parameters for the local units are given in Table 6.2). Blue depicts the predictive strategy and green depicts the optimal strategy.
Notice that the optimal strategy has a better utilization of e.g. LU_7 and LU_9 early in the simulations. This allows the optimal strategy to stay clear of $\overline{E}$ for all local units until sample 70.

minimum runtime constraints to the local unit models. When such temporal constraints are added it seems unlikely that an optimal strategy can be found for any future trajectory of $P_{Dispatch}$. It might, however, be possible to formulate general strategies or rules-of-thumb for these models even without the assumption of prediction. This would be highly advantageous for large scale problems where computational demands become significant.

References

[1] Kai Heussen, Stephan Koch, Andreas Ulbig, Göran Andersson, Energy Storage in Power System Operation: The Power Nodes Modeling Framework, IEEE PES Conference on Innovative Smart Grid Technologies Europe, 2010, pp. 1-8.

[2] M. Kraning, Y. Wang, E. Akuiyibo, S. Boyd, Operation and Configuration of a Storage Portfolio via Convex Optimization, Proceedings IFAC World Congress, Milan, August 2011

[3] Shi You, Chresten Træholt, Bjarne Poulsen (2009), Generic Virtual Power Plants: Management of Distributed Energy Resources under Liberalized Electricity Market, The 8th IET International Conference on Advances in Power System Control, Operation and Management, Hong Kong, P.R. China, 2009

[4] Klaus Trangbæk, Jan Bendtsen, Mette Petersen and Jakob Stoustrup (2011), Exact Power Constaints in Smart Grid Control, Proceedings of the 50th IEEE Conference on Decision and Control and European Control Conference, Orlando, December 2011

[5] N.S. Wade, P.C. Taylor, P.D. Lang and P.R. Jones, Evaluating the Benefits of an Electrical Energy Storage System in a Future Smart Grid, Energy Policy - 2010, Volume 38, pp. 7180–7188

[6] A. Mohsenian-Rad, V.W.S. Wong, J. Jatskevich and R. Schober and A. Leon-Garcia, Autonomous Demand-side Management Based on Game-theoretic Energy Consumption Scheduling for the Future Smart Grid, IEEE Transactions on Smart Grid - 2010, Volume 1, Issue 3, pp. 320–331

[7] Ivar Wangensteen (2007), Power System Economics - the Nordic Electricity Market. Tapir Academic Press.

[8] Steven Stoft (2002), Power System Economics Designing Markets for Electricity. Wiley-IEEE Press.

[9] Energinet.dk (2011), Systemydelser til levering i Danmark, Udbudsbetingelser.

Paper C

A Taxonomy for Modeling Flexibility and a Computationally Efficient Algorithm for Dispatch in Smart Grids

Mette Kirschmeyer Petersen, Kristian Edlund, Lars Henrik Hansen, Jan Dimon Bendtsen and Jakob Stoustrup

This paper was published in: Proceedings of American Control Conference, 2013.

The layout has been revised

Abstract

The word flexibility is central to Smart Grid literature, but to this day a formal definition of flexibility is still pending. This paper present a taxonomy for modeling flexibility in Smart Grids, denoted *Buckets*, *Batteries and Bakeries*.

We consider a direct control Virtual Power Plant (VPP), which is given the task of servicing a portfolio of flexible consumers by use of a fluctuating power supply. Based on the developed taxonomy we first prove that no *causal* optimal dispatch strategies exist for the considered problem. We then present two heuristic algorithms for solving the balancing task: *Predictive-Balancing* and *Agile-Balancing*.

Predictive-Balancing, is a traditional moving horizon algorithm, where power is dispatched based on perfect predictions of the power supply. *Agile-Balancing*, on the other hand, is strictly non-predictive. It is, however, explicitly designed to exploit the heterogeneity of the flexible consumers.

Simulation results show that in spite of being non-predictive *Agile-Balancing* can actually out-perform *Predictive-Balancing* even when *Predictive-Balancing* has perfect prediction over a relatively long horizon. This is due to the flexibility-synergy-effects, which *Agile-Balancing* generates. As a further advantage it is demonstrated, that *Agile-Balancing* is extremely computationally efficient since it is based on sorting rather than linear programming.

1 Introduction

The introduction of renewable energy production into the existing power system is complicated by the inherent variability of production technologies, which harvest energy mainly from wind and sun. This means that it becomes increasingly challenging to maintain the real-time balance between production and consumption as the ratio of renewable energy production increases. In a Smart Grid system, on the other hand, the inherent flexibility of consumers, such as electric vehicles, heat pumps and HVAC-systems, may be mobilized to play an active part in solving the balancing task.

The flexibility of a given system is a unique, innate, state- and time dependent quality. In conversation it is therefore sometimes said that *flexibility is the ability to deviate from the plan*. That characterization of flexibility is very insightful, but it still leaves us with the problem of defining both *the ability to deviate* and *the plan*.

In this paper we focus on *the ability to deviate* by proposing a taxonomy for modeling flexibility. The numerous constraints that characterize a given flexible system were first investigated in [19]; in the present paper, however, we have chosen to focus on the constraints of

I) Power Capacity,

II) Energy Capacity,

III) Energy level at a specific deadline, and

IV) Minimum runtime,

since these are widely found in practical systems.

Our taxonomy is denoted *Buckets*, *Batteries and Bakeries* and precise definitions are given in Section 4. *The Bucket*, *The Battery* and *The Bakery* are three simple flexibility

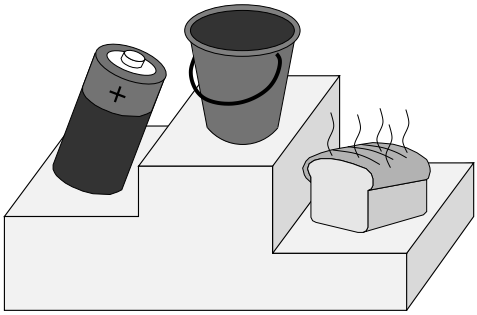

Figure 7.1: *Buckets*, *Batteries and Bakeries* is a taxonomy for modeling flexibility in Smart Grids.

models, which are constructed based on the constraints I) to IV). The first model, denoted *the Bucket*, is a power and energy constrained integrator. The *Bucket* could be used as a simplified model of a house with a heat pump, which is used for energy storage. The *Battery* is also a power and energy constrained integrator, but with the added restriction that the unit must be fully charged at a specific deadline. The *Battery* could be modeling an electric vehicle, which must be ready for operation at a specific time. Finally the *Bakery* extends the *Battery* with the additional constraint that the process must run in one continuous stretch at constant power consumption. The *Bakery* could be a commercial green house, where plants must recieve a specific amount of light each day. This light must, however, be delivered continuously to stimulate the photosynthesis of the plants.

The suggested framework is a proper taxonomy in the sense that we have imposed a hierarchical relationship between the three models. This means that *a Bucket* provides a better quality of flexibility than *a Battery*, which is again superior to *a Bakery* (see Figure 7.1). Here, better quality means less restricted, not necessarily more flexible. The reason for this distinction is that the flexibility of a system is not just determined by constraints, but also by the specific parameter values of the system. That is, a "large" *Battery* could therefore be said to be more flexible than a "small" *Bucket*, even though the *Bucket* is a better quality flexibility than the *Battery*.

Based on the hierarchical relationship between models we will develop an algorithm, *Agile-Balancing*, which exploits the heterogeneity of flexible systems. This makes *Agile-Balancing* robust against prediction errors and computationally efficient at the same time.

The paper is structured as follows: First, Section 2 gives an extensive review of how flexibility is modeled in Smart Grid literature today. Next, Section 3 and 4 present the considered optimization problem and the taxonomy. Following this, it is proved formally in Section 5 how *causality* [16] relates to the taxonomy. Finally, Section 6 and 7 present *Predictive-Balancing* and *Agile-Balancing* and give comparative simulation examples.

2 State-of-the-Art

A review of how flexibility is modeled in Smart Grid literature reveals that the generic models of *Buckets*, *Batteries* and *Bakeries* are certainly not novel concepts. Several works

Reference	Perfect prediction	*Bucket*	*Battery*	*Bakery*
[1]	Yes	x		
[2]	Yes	x		
[3]	No	x		
[4]	Yes	x		
[5]	No	x		
[6]	No	x	(x)	
[7]	Yes	x	x	
[8]	No	x		(x)
[9]	No	(x)	(x)	(x)
[10]	Yes		x	
[11]	Yes		x	
[12]	Yes		x	
[13]	Yes[1]		x	
[14]	No		x	
[15]	Yes		x	
[16]	No		x	x
[17]	Yes		x	x
[18]	Yes			x

Table 7.1: Review of flexibility modeling in Smart Grid literature.

have been identified (see Table 7.1), which model flexibility in ways very similar to *a Bucket*, *a Battery* or *a Bakery*. Most existing literature, however, focuses on optimized operation of one particular technology. This means that the advantages of heterogeneity are not investigated.

In [9] a modeling framework for demand response technologies is formulated based on Markov Chain processes. This framework has some similarity to the taxonomy suggested in the present work. The authors of [9] subscribe to the concept of price-signalling, however; possible synergies between heterogenous subsystems are therefore not investigated, since these can only really be exploited though direct control.

The work closest related to the concepts investigated in this paper, is [16]; in fact, the term *laxity*, as used in [16], is almost synonymous with the term *agility* used in [5]. Only the *Battery*-model is investigated [16], however.

In our literature review we have also charted the use of the assumption of perfect prediction[1], which is found to be quite widespread.

3 Problem Formulation

Consider a Virtual Power Plant, which must provide power to a portfolio of flexible systems by dispatching a fluctuating power supply. The fluctuating power supply is denoted $P_{Dispatch}(k), k = 1, 2, \ldots, K$, and the flexible systems are denoted local units. A port-

[1] Paper [13] does assume perfect prediction as indicated in Table 7.1, but the effects of uncertainty are also investigated.

folio of N local units is denoted $\{LU_i\}_{i=1,2,...,N}$. At sample k we let $P_i(k)$ denote the power, which is dispatched to unit i, and any quantity, which cannot be dispatched to the portfolio, is denoted $\mathcal{S}(k)$. The objective is to minimize the residual power, that is $|\mathcal{S}|$.

The problem can be formulated as

$$\min_{P_i(\cdot)} \sum_{k=0}^{\infty} |\mathcal{S}(k)| \tag{7.1}$$

$$s.t.$$

$$P_{Dispatch}(k) \in \mathbb{R}, k = 0, 1, ..., \infty \tag{7.2}$$

$$\sum_{i=1}^{N} P_i(k) + \mathcal{S}(k) = P_{Dispatch}(k) \tag{7.3}$$

and also subject to the dynamics and constraints of $\{LU_i\}_{i=1,2,...,N}$.

4 Taxonomy: Buckets, Batteries and Bakeries.

This section defines the *Buckets*, *Batteries and Bakeries-taxonomy* for modeling flexibility in Smart Grids.

Formal definitions of a *Bucket*, a *Battery* and a *Bakery* are given in Definition 15, 16 and 17 respectively, and the models are further illustrated in Figure 7.2, 7.3 and 7.4. In the following T_s denotes the size of the time step, $\underline{P}_i$ and $\overline{P}_i$ denote limits on consumption rate, $\underline{E}_i$ and $\overline{E}_i$ denote limits on energy storage levels and $v_i(k)$ is a boolean-valued variable stating whether or not a *Bakery* is running at sample k.

Definition 15 (Bucket)**.** The dynamics and constraints of a *Bucket* are

$$\begin{aligned} \textit{Bucket}_i(k)\text{: } & E_i(k+1) = E_i(k) + T_s P_i(k) \\ & \underline{P}_i \leq P_i(k) \leq \overline{P}_i \\ & \underline{E}_i \leq E_i(k) \leq \overline{E}_i \\ & E_i(0) = E_{i,0}, \end{aligned}$$

where $k = 0, 1, \ldots, \infty$, $i = 1, 2, \ldots, N^{\textit{Buckets}}$, $\underline{P}_i \leq 0 \leq \overline{P}_i$ and $\underline{E}_i \leq E_{i,0} \leq \overline{E}_i$.

Definition 16 (Battery)**.** The dynamics and constraints of a *Battery* are

$$\begin{aligned} \textit{Battery}_i(k)\text{: } & E_i(k+1) = E_i(k) + T_s P_i(k) \\ & 0 \leq P_i(k) \leq \overline{P}_i \\ & 0 \leq E_i(k) \leq \overline{E}_i \\ & E_i(0) = E_{i,0}, \\ & E_i(T_{end,i}) = \overline{E}_i, \end{aligned}$$

where $k = 0, 1, \ldots, \infty$, $i = 1, 2, \ldots, N^{\textit{Batteries}}$, $T_{end,i} \in \mathbb{N}$, $0 \leq \overline{P}_i$ and $0 \leq \overline{E}_i$.

Definition 17 (Bakery)**.** The dynamics and constraints of a *Bakery* are

$$
\begin{aligned}
\textit{Bakery}_i(k)\text{: } & E_i(k+1) = E_i(k) + T_s P_i(k), \\
& P_i(k) = \overline{P}_i v_i(k) \\
& 0 \leq E_i(k) \leq \overline{E}_i, \\
& E_i(0) = E_{i,0}, \\
& E_i(T_{end,i}) = \overline{E}_i, \\
& 0 \leq \sum_{l=k}^{k+T_{run,i}-1} v_i(l) - T_{run,i}\Big(v_i(k) - v_i(k-1)\Big),
\end{aligned}
$$

where $k = 0, 1, \ldots, \infty$, $0 \leq \overline{P}_i$, $\overline{E}_i = \overline{P}_i T_{run,i}$, $v_i(k) \in \{0, 1\}$, $i = 1, 2, \ldots, N^{\textit{Bakeries}}$, $T_{end,i} \in \mathbb{N}$ and $T_{run,i} \in \mathbb{N}$.

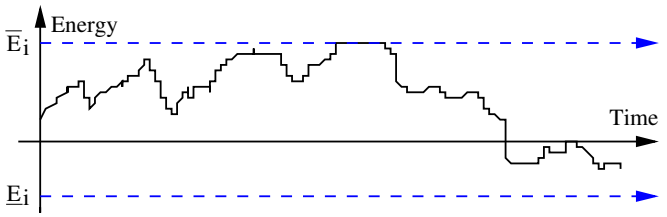

Figure 7.2: A *Bucket* is a power and energy constrained integrator.

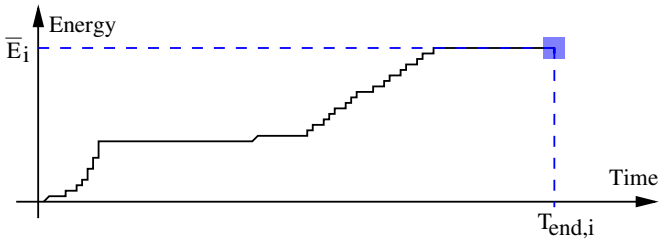

Figure 7.3: A *Battery* is a power and energy constrained integrator, which must be "charged" to level $\overline{E}_i$ by time $T_{end,i}$.

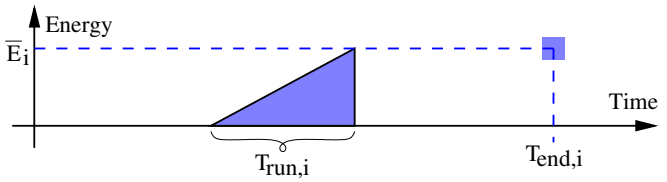

Figure 7.4: A *Bakery* is a batch process, which must be finished by time $T_{end,i}$. The process has constant power consumption and the run time is $T_{run,i}$.

5 Causality

In [16] a dispatch strategy was defined as *causal* if it depends only on the information state at time k. The authors of [16] also proved that an optimal *causal* dispatch strategy does not exist for a portfolio of *Batteries*. It was shown in [5] that adding the constraint $\underline{P} = \underline{E} = 0$ for a portfolio of *Buckets* induces that an optimal *causal* dispatch strategy *does* exist. For the sake of completion this section will prove that an optimal *causal* dispatch strategy does not, in general exist for a portfolio consisting of only *Buckets* or only *Bakeries*.

Proposition 1. There does not exist an optimal *causal* dispatch strategy for a portfolio of *Buckets*.

Proof. Proof is done by counterexample. Consider a portfolio consisting of the following two *Buckets*

$$\begin{aligned} \textit{Bucket}_1\colon\ & E_1(0) = 0, \\ & \overline{P}_1 = 1, \overline{E}_1 = 1, \\ & \underline{P}_1 = -1, \underline{E}_1 = -1, \end{aligned}$$

$$\begin{aligned} \textit{Bucket}_2\colon\ & E_2(0) = 0, \\ & \overline{P}_2 = 1, \overline{E}_2 = 3, \\ & \underline{P}_2 = -1, \underline{E}_2 = -3, \end{aligned}$$

Next define the following dispatch profiles

$$\begin{aligned} P^A_{Dispatch} &= (0, 2, 2), \\ P^B_{Dispatch} &= (0, -2, -2). \end{aligned}$$

Observe that it is possible to dispatch sequence $P^A_{Dispatch}$ in such a way that $\sum_{k=0}^{2} |S| = 0$. However, this is only achievable if $P_1(0) = -1$ and $P_2(0) = 1$. Observe also that equivalent arguments hold for $P^B_{Dispatch}$ if $P_1(0) = 1$ and $P_2(0) = -1$. At $k = 0$ a *causal* dispatch strategy must offer allocations based only on information available at time $k = 0$. Notice, however, that $P^A_{Dispatch}(0) = P^B_{Dispatch}(0)$ and since optimal dispatch of $P^A_{Dispatch}$ and $P^B_{Dispatch}$ requires different allocations at time $k = 0$, a *causal* dispatch strategy cannot exist. □

Proposition 2. There does not exist an optimal *causal* dispatch strategy for a portfolio of *Bakeries*.

Proof. Proof is done by counterexample. Consider a portfolio consisting of the following two *Bakeries*

$$\begin{aligned} \textit{Bakery}_1\text{: } & E_1(0) = 0, \\ & \overline{P}_1 = 1, \overline{E}_1 = 1, \\ & T_{run,1} = 1, T_{end,1} = 2, \\ \textit{Bakery}_2\text{: } & E_2(0) = 0, \\ & \overline{P}_2 = 3, \overline{E}_2 = 3, \\ & T_{run,2} = 1, T_{end,2} = 2. \end{aligned}$$

Next define the following dispatch profiles

$$\begin{aligned} P^A_{Dispatch} &= (2, 1), \\ P^B_{Dispatch} &= (2, 3). \end{aligned}$$

Observe that the optimal dispatch of either sequence $P^A_{Dispatch}$ or sequence $P^B_{Dispatch}$ to the portfolio has $\sum_{k=0}^{1} |S| = 1$. However, for $P^A_{Dispatch}$, this is only achievable if $P_1(0) = 0$ and $P_2(0) = 3$. For $P^B_{Dispatch}$ the required configuration is $P_1(0) = 1$ and $P_2(0) = 0$. The argumentation that a *causal* optimal dispatch strategy does not exist now follows as in the proof of Proposition 1. □

6 Algorithms

Since we have proven that *causal* optimal dispatch strategies do not exist, this section will present two heuristic algorithms for solving problem (7.1) - (7.3). The algorithms are denoted *Predictive-Balancing* and *Agile-Balancing*.

Predictive-Balancing

A strategy for solving problem (7.1) - (7.3) is to use a moving horizon approach. To do this, we assume perfect prediction of $P_{Dispatch}$ over a certain prediction horizon K, and solve

$$\min_{P_i(\cdot)} \sum_{k=1}^{K} w_k |\mathcal{S}(k)| \tag{7.4}$$

$$s.t.$$

$$P_{Dispatch}(k) \in \mathbb{R}, \tag{7.5}$$

$$\sum_{i=1}^{N} P_i(k) + \mathcal{S}(k) = P_{Dispatch}(k), \tag{7.6}$$

where $w_{k_1} > w_{k_2}$ if $k_1 < k_2$. Adding the impatience weights w_k to the cost function ensures that if the problem cannot be solved without introducing slack, then the imbalances will incur as late within the prediction horizon as possible.

Agile-Balancing

The main objective of the present paper is to investigate heterogenous systems and we do this by introducing agility factors for each class of flexibility. The agility factor of a given unit should express the quality (see [19]) of the flexibility, which the unit represents.

The authors of the present paper first investigated the agility attributes of the *Bucket*-model in [5]. Here agility factors for the *Bucket*-model were defined as

Definition 18 (Agility Factor, Bucket)**.**
Let $\textit{Bucket}_i(k)$ denote a *Bucket*. The *agility factor* of *Bucket* i at sample k is

$$\mathcal{K}_i^{\textit{Bucket}}(k) = \frac{\overline{E}_i - E_i(k)}{T_s \overline{P}_i}.$$

With this definition of the agility factor for the *Bucket*-model we obtain that $\mathcal{K}_i^{\textit{Bucket}}(k)$ denotes the number of samples that the *Bucket* can operate at maximum power without becoming inactive/full.

Introducing *Batteries* and *Bakeries* to the portfolio means that in addition to balancing $P_{Dispatch}$ the Virtual Power Plant must solve a set of fixed tasks, namely charging the *Batteries* and starting the *Bakeries* in due time. This means that as a deadline, T_{end}, approaches, a *Battery* or a *Bakery* can go from being a flexible resource, which can help to minimize our objective, to being a constraint. We therefore define agility factors for the *Battery*- and *Bakery* models, which state how close we are (in terms of samples) to being forced to charge a battery or start bakery:

Definition 19 (Agility Factor, Battery)**.**
Let $\textit{Battery}_i(k)$ denote a *Battery*. The *agility factor* of *Battery* i at sample k is

$$\mathcal{K}_i^{\textit{Battery}}(k) = T_{end,i} - k - \frac{\overline{E}_i - E_i(k)}{T_s \overline{P}_i}.$$

Definition 20 (Agility Factor, Bakery)**.**
Let $\textit{Bakery}_i(k)$ denote a *Bakery*. The *agility factor* of *Bakery* i at sample k is

$$\mathcal{K}_i^{\textit{Bakery}}(k) = T_{end,i} - T_{run,i} - k.$$

Notice that the definition of agility factors for the *Battery* is the same as the definition of a flexibility factors used in [16].

As the deadline of a *Battery* or a *Bakery* approaches the Virtual Power Plant can be forced to charge that *Battery* or start that *Bakery* irrespective of whether this is beneficial to its objective. Forced consumption on LU_i at sample k can, however, be computed based on the agility factors, as

$$P_{Forced,i}^{\textit{Battery}}(k) = \begin{cases} 0 & \mathcal{K}_i^{\textit{Battery}} > 1 \\ \overline{P}_i(1 - \mathcal{K}_i^{\textit{Battery}}) & 1 \geq \mathcal{K}_i^{\textit{Battery}} > 0 \\ \overline{P}_i & \mathcal{K}_i^{\textit{Battery}} = 0 \end{cases}$$

and

$$P_{Forced,i}^{\textit{Bakery}}(k) = \begin{cases} 0 & \mathcal{K}_i^{\textit{Bakery}} > 1 \\ \overline{P}_i & \mathcal{K}_i^{\textit{Bakery}} = 0. \end{cases}$$

The algorithm *Agile-Balancing* is based on the principle of *flexibility maximization* [19], where the worst quality units are dispatched first at each sample. The idea is simple: At each sample the Virtual Power Plant will first focus on the set assignments of charging *Batteries* and starting *Bakeries* . The Virtual Power Plant will solve the most pressing task first and the unit with the smallest agility factor is the most critical asset in need of service. At sample k *Agile-Balancing* therefore dispatches as much power as possible to the *Batteries* and *Bakeries*, but no more than $P_{Dispatch}(k)$. Secondly, *Agile-Balancing* uses the buffer available in the *Buckets* to minimize any remaining imbalance.

Since there are no energy requirements on a *Bucket*, it can only constitute a resource and never a constraint. There are both power and energy constraints on a *Bucket*, however, meaning that only a limited amount of power can be dispatched to the *Bucket*-portion of the portfolio at each sample. The maximum amount of power, which can be dispatched to *Bucket*$_i$ at sample k is denoted $P^{Bucket}_{Reserve,i}(k)$ and is given as

$$P^{Bucket}_{Reserve,i}(k) = \min\left(\overline{P}_i, \frac{\overline{E}_i - E_i(k)}{T_s}\right).$$

At sample k the upper reserve bound on a portfolio containing $N^{Buckets}$ *Buckets* is therefore

$$P^{Bucket}_{Reserve}(k) = \sum_{i=1}^{N^{Buckets}} \min\left(\overline{P}_i, \frac{\overline{E}_i - E_i(k)}{T_s}\right).$$

Furthermore, *Agile-Balancing* handles any dispatch to *Buckets* by implementing the linear cost function given in [5]. Pseudo-code for *Agile-Balancing* is given in Algorithm 4.

Algorithm 4 :
Agile Balancing $\Big(\{\boldsymbol{LU}_i\}_{\boldsymbol{i=1,2,...,N}}, \boldsymbol{P_{Dispatch}}\Big)$

for $k = 1$ **to** K **do**

Compute $P_{Forced}(k) = \sum_{i=1}^{N^{Batteries}} P^{Batteries}_{Forced,i}(k) + \sum_{j=1}^{N^{Bakeries}} P^{Bakeries}_{Forced,j}(k)$.

if $P_{Forced}(k) > P_{Dispatch}(k)$ **then**

$P^{Batteries}(k) = P^{Batteries}_{Forced}(k)$,

$P^{Bakeries}(k) = P^{Bakeries}_{Forced}(k)$.

else

Sort *Batteries* and *Bakeries* according to increasing agility factor.

Distribute $P_{Dispatch}(k)$ to *Batteries* and *Bakeries* in increasing agility factor order and such that $P^{Batteries}(k) + P^{Bakeries}(k)$ is as large as possible, but less than or equal to $P_{Dispatch}(k)$.

end if

Define $P^{Buckets}(k) = \min\Big(P^{Buckets}_{Reserve}(k), P_{Dispatch}(k) - P^{Batteries}(k) - P^{Bakeries}(k)\Big)$.

Distribute $P^{Buckets}(k)$ to the *Buckets* as prescribed in [5] that is in decreasing agility factor order.

Set $S(k) = P_{Dispatch}(k) - P^{Buckets}(k) - P^{Batteries}(k) - P^{Bakeries}(k)$.

end for

7 Simulation Examples

This section presents two simulation examples. The first simulation example compares the performance of *Predictive-Balancing* and *Agile-Balancing*. The second simulation example investigates the computational efficiency of *Agile-Balancing*. In all simulations we have $T_s = 1$ and $E_{i,0} = 0$ for all units. Solutions of problem (7.4) - (7.6) are computed by use of CPlex, [20]. *Agile-Balancing* has been implemented in C#. Computations are performed on a standard laptop.

Predictive-Balancing vs. Agile-Balancing

This simulation example considers a randomly generated portfolio of 105 units, where $N^{Buckets} = 5$ and $N^{Batteries} = N^{Bakeries} = 50$. All units have $\frac{\overline{E}}{T_s\overline{P}} \leq 10$ and $\sum_{Portfolio} \overline{E} = 50$.

The results of running *Predictive-Balancing* for $K = 10$ are given in Figure 7.5. When there is a drop in $P_{Dispatch}$ *Predictive-Balancing* attempts to use the *Buckets* as

buffer to maintain the balance between supply and demand. Towards the end of each low-period, however, *Predictive-Balancing* is forced to use significant slack. This occurs because the prediction horizon is not sufficiently long, and the problem could be mended by increasing the prediction horizon. However, such a modification comes at the price of computation time, which we will explore later in this section.

The results of running *Agile-Balancing* are presented in Figure 7.6. When there is a drop in the power supply *Agile-Balancing* is poorly prepared and therefore has too many *Bakeries* started. Since the *Bakeries* cannot be shut down *Agile-Balancing* must utilize the buffer in the *Buckets* to maintain the balance. With the given portfolio *Agile-Balancing* is able to balance supply and demand without introducing slack until the very end of the simulation.

Computation times and the sum of the absolute value of the slack variable are given in Table 7.2 for $K = 10$, $K = 15$ and $K = 20$. Notice that *Predictive-Balancing* must have perfect prediction of at least 20 samples to perform better than *Agile-Balancing*. As the prediction horizon increases, so does the computation time of *Predictive-Balancing*, however; notice that even with a prediction horizon of only 10 samples, *Predictive-Balancing* is almost one hundred times slower than *Agile-Balancing*. This is because the most computationally demanding task *Agile-Balancing* must solve is to sort units according to agility factor. *Predictive-Balancing*, on the other hand, solves a series of mixed integer programs, which is far more computationally demanding.

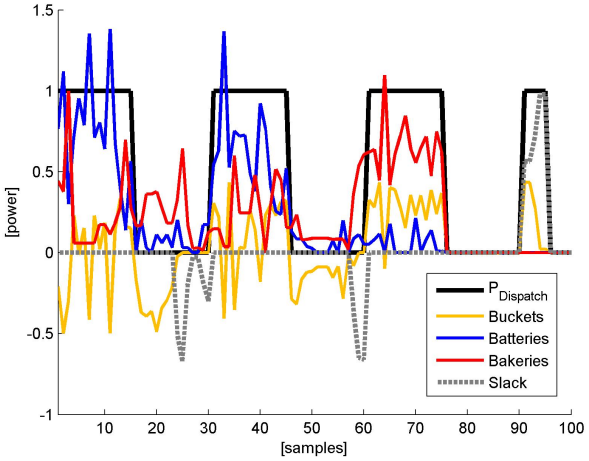

Figure 7.5: Power dispatched at each sample for each type of unit by *Predictive-Balancing* when $K = 10$.

Large Scale Simulations

This simulation example further investigates the computational efficiency of *Agile-Balancing* by considering a randomly generated portfolio of 10^6 units. All units have $\frac{\overline{E}}{T_s \overline{P}} \leq 30$.

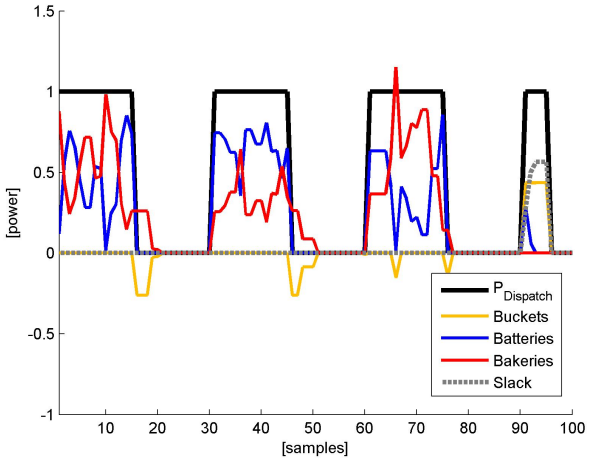

Figure 7.6: Power dispatched at each sample for each type of unit by *Agile-Balancing*.

	Comp. Time [s]	$\sum \lvert\mathcal{S}(\cdot)\rvert$
Agile-Balancing	0.03	2.48
Predictive-Balancing, K = 10	2.5	7.40
Predictive-Balancing, K = 15	4.0	4.29
Predictive-Balancing, K = 20	5.8	1.92

Table 7.2: Computation time and the sum of numerical imbalances for *Predictive-Balancing* and *Agile-Balancing*.

Dyn. Ag.	*Buckets*	*Batteries*	*Bakeries*	Comp. Time	$\sum \lvert\mathcal{S}(\cdot)\rvert$
Yes	33%	33%	33%	3 min. 26 sec.	0
Yes	10%	45%	45%	3 min. 25 sec.	19712
No	33%	33%	33%	1 min. 1 sec.	0
No	10%	45%	45%	1 min. 4 sec.	43264

Table 7.3: Computation time and the sum of numerical imbalances for large scale simulation.

Figure 7.7 depicts the simulation results, when one third of each type of unit is included in the portfolio and in Figure 7.8 only 10% *Buckets* are included in the portfolio. Computation times and the sum of the absolute value of imbalances are given in Table 7.3. In Smart Grid discussions it is often proposed that if only the number of units under the jurisdiction of a Virtual Power Plant is large enough, then *the-law-of-big-numbers* will ensure that the aggregated behavior of the portfolio will be the same as that of a traditional power plant (so essentially proposing that a large portfolio will exhibit *Bucket*-behavior).

However, the second simulation (Figure 7.8) is an example of a case where a large number of units is not in itself enough to warrant that the load can be balanced. This illustrates that care must be taken the ensure that the right combination of units is available in the portfolio.

To further improve the computation time *Agile-Balancing* has also been implemented without using dynamic agility factors. This means modifying Algorithm 4 by moving line 7 to the very start of the algorithm (before the for-loop), such that only one sorting is performed. The results of these simulations are given in Figure 7.9, Figure 7.10 and Table 7.3. As expected, sorting only once per simulation gives a significant speed up of the computation time, as the modified implementation is more than three times faster than the original. With a portfolio of one third of each type of units, there is no cost of this speed up in terms of performance/optimality. With only 10% *Buckets* in the portfolio, however, not having dynamic agility factors has a significant cost in terms of performance.

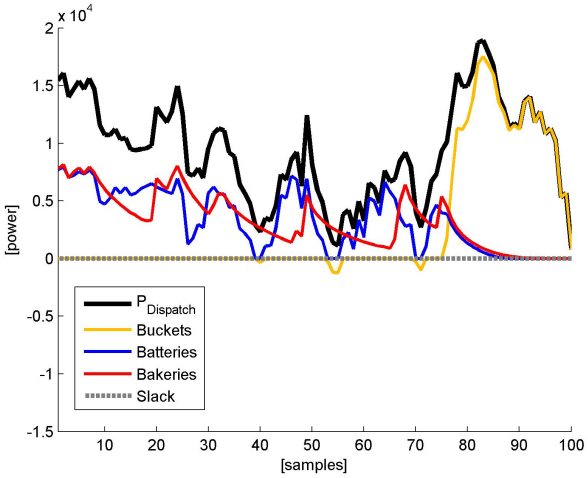

Figure 7.7: Power dispatched at each sample for each type of unit by *Agile-Balancing* for a portfolio of 1.000.000 units having one third of each type.

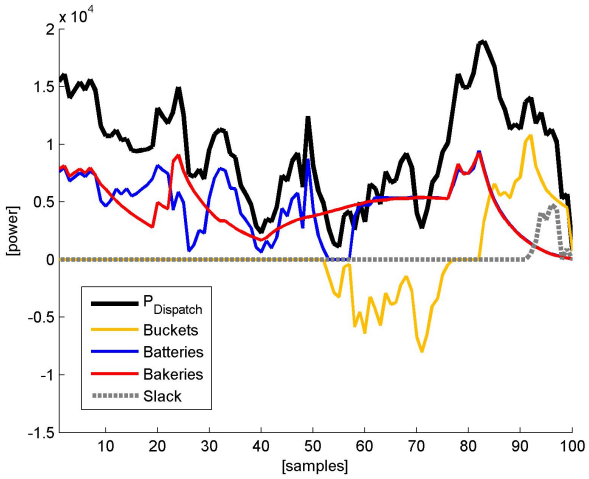

Figure 7.8: Power dispatched at each sample for each type of unit by *Agile-Balancing* for a portfolio of 1.000.000 units with 10% *Buckets*, 45% *Batteries* and 45% *Bakeries*.

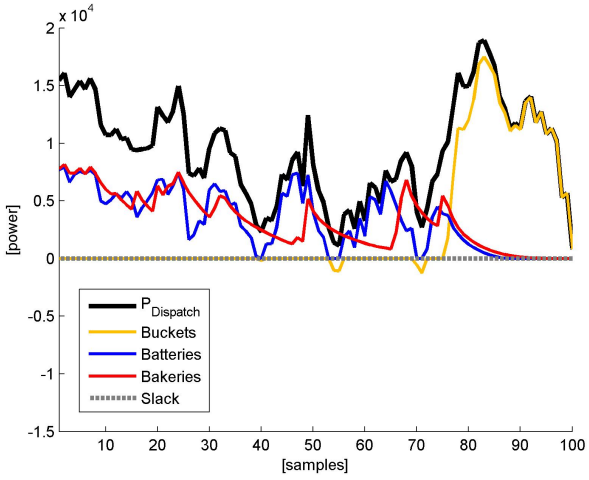

Figure 7.9: Power dispatched at each sample for each type of unit by *Agile-Balancing* for a portfolio of 1.000.000 units having one third of each type and not using dynamic agility factors.

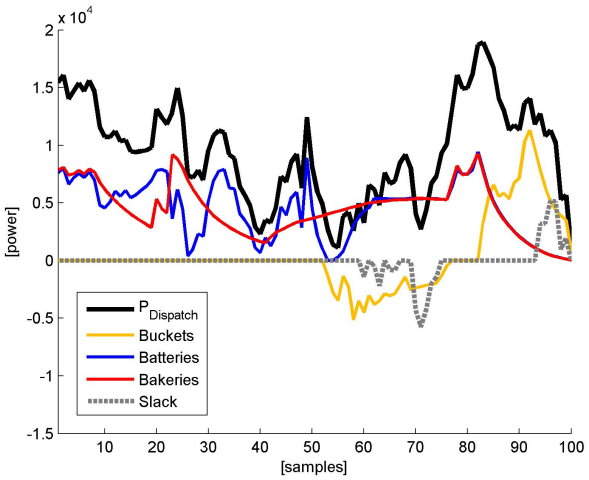

Figure 7.10: Power dispatched at each sample for each type of unit by *Agile-Balancing* for a portfolio of 1.000.000 units with 10% *Buckets*, 45% *Batteries* and 45% *Bakeries* and not using dynamic agility factors.

8 Conclusion

In this paper we have identified a number of common traits shared by most, if not all, power consuming or -producing units that can be expected to appear in a future Smart Grid system. Most literature to date has focused on only one type of units or one particular technology, although some references have treated more than one type. We proposed a taxonomy that allows the division of units into three distinct categories based on key traits of the unit's primary purpose such as minimum runtime, the ability to consume/release power back to the grid, minimum consumption by a certain time, etc., in a quantifiable manner.

We have also presented a suboptimal, but extremely computationally efficient dispatch algorithm, denoted *Agile-Balancing*. One of the main challenges in developing the Smart Grid is the sheer size of optimization problems involved. This means that the computation time associated with determining optimal solutions might be unacceptable in practice. An optimal solution available two minutes after market gate closure is far less useful than a suboptimal one available two minutes before market gate closure; thus, even though *Agile-Balancing* is not optimal, it might still be the best solution in practice.

References

[1] Kai Heussen, Stephan Koch, Andreas Ulbig, Göran Andersson, *Energy Storage in Power System Operation: The Power Nodes Modeling Framework*, IEEE PES Conference on Innovative Smart Grid Technologies Europe, 2010, pp. 1-8.

[2] Benjamin Biegel, Jakob Stoustrup, Jan Bendtsen and Palle Andersen, *Model Predictive Control for Power Flows in Networks with Limited Capacity*, 2012 American Control Conference, 2012, pp. 2959-2964.

[3] Ali Faghih, Mardavij Roozbehani and Munther A. Dahleh, *Optimal Utilization of Storage and the Induced Price Elasticity of Demand in the Presence of Ramp Constraints*, 50th IEEE Conference on Decision and Control and European Control Conference, 2011, pp. 842-847.

[4] T.Y. Lee and N. Chen, *Effect of the Battery Energy Storage System on the Time Of Use Rates Industrial Customers*, IEE Proc.-Gener. Transm. Distrib., Vol 141, No. 5, September 1994.

[5] Mette Petersen, Jan Dimon Bendtsen and Jakob Stoustrup, *Optimal Dispatch Strategy for the Agile Virtual Power Plant*, 2012 American Control Conference, 2012, pp. 288-294.

[6] Matt Kraning, Yang Wang, Ekine Akuiyibo, Stephen Boyd, *Operation and Configuration of a Storage Portfolio via Convex Optimization*, 18th IFAC World Congress, 2011, pp. 10487-10492.

[7] Nikolaos Gatsis and Georgios B. Giannakis, *Residential Demand Response with Interruptible Tasks: Duality and Algorithms*, 50th IEEE Conference on Decision and Control and European Control Conference, 2011, pp. 1-6.

[8] Ioannis Ch. Paschalidis, Binbin Li, Michael C. Caramanis, *A Market-Based Mechanism for Providing Demand-Side Regulation Service Reserves*, 50th IEEE Conference on Decision and Control and European Control Conference, 2011, pp. 1-6., pp. 21-26.

[9] Konstantin Turitsyn, Scott Backhaus, Maxim Ananyev and Michael Chertkov, *Smart Finite State Devices: A Modeling Framework for Demand Response Technologies*, 50th IEEE Conference on Decision and Control and European Control Conference, 2011, pp. 7-14.

[10] B. Daryanian, R.E. Bohn and R.D. Tabors, *Optimal Demand-Side Response to Electricity Spot Proices for Storage-Type Customers*, IEEE Transactions on Power Systems, Vol. 4, No. 3, 1989.

[11] Amir-Hamed Mohsenian-Rad, Vincent W. S. Wong, Juri Jatskevich, Robert Schober and Alberto Leon-Garcia, *Autonomous Demand-Side Management Based on Game-Theoretic Energy Consumption Scheduling for the Future Smart Grid*, IEEE Transactions on Smart Grid, Vol. 1, No. 3, 2010.

[12] Angel Rosso, Juan Ma, Daniel S. Kirschen and Luis F. Ochoa, *Assessing the Contribution of Demand Side Management to Power System Flexibility*, 50th IEEE Conference on Decision and Control and European Control Conference, 2011, pp. 4361-4365.

[13] Changsun Ahn, Chiao-Ting Li and Huei Peng, *Decentralized Charging Algorithm for Electrified Vehicles Connected to Smart Grid*, American Control Conference, 2011.

[14] Anthony Papavasiliou and Shmuel S. Oren, *Supplying Renewable Energy to Deferrable Loads: Algorithms and Economic Analysis*, IEEE Power and Energy Society General Meeting, 2010.

[15] Ralph Hermans, Mads Almassalkhi and Ian Hiskens, *Incentive-based Coordinated Charging Control of Plug-in Electric Vehicles at the Distribution-Transformer Level*, 2012 American Control Conference, 2012, pp. 264-269.

[16] A. Subramanianz, M. Garcia, A. Domnguez-Garca, D. Callaway, K. Poollay and P. Varaiyay, *Real-time Scheduling of Deferrable Electric Loads*, 2012 American Control Conference, 2012, pp. 3643-3650.

[17] Jing Huang, Vijay Gupta and Yih-Fang Huang, *Scheduling Algorithms for PHEV Charging in Shared Parking Lots*, 2012 American Control Conference, 2012, pp. 276-281.

[18] Kin Cheong Sou, James Weimer, Henrik Sandberg, and Karl Henrik Johansson, *Scheduling Smart Home Appliances Using Mixed Integer Linear Programming*, 50th IEEE Conference on Decision and Control and European Control Conference, 2011.

[19] Mette Petersen, Lars Henrik Hansen and Tommy Mølbak, *Exploring the Value of Flexibility: A Smart Grid Discussion*, 8th IFAC Conference on Power Plant and Power System Control, 2012.

[20] `www-01.ibm.com/software/integration/optimization/cplex-optimization-studio/`

Paper D

Market Integration of Virtual Power Plants

Mette Kirschmeyer Petersen, Lars Henrik Hansen, Jan Dimon Bendtsen, Kristian Edlund and Jakob Stoustrup

This paper was published in: Proceedings of the 52nd IEEE Conference on Decision and Control, 2013.

Abstract

We consider a direct control Virtual Power Plant, which is given the task of maximizing the profit of a portfolio of flexible consumers by trading flexibility in Energy and Power Markets. Spot price optimization has been quite intensively researched in Smart Grid literature lately. In this work, however, we develop a three stage market model, which includes Day-Ahead (Spot), Intra-Day and Regulating Power Markets. This allows us to test the hypothesis that the Virtual Power Plant can generate additional profit by trading across several markets.

We find that even though profits do increase as more markets are penetrated, the size of the profit is strongly dependent on the type of flexibility considered. We also find that penetrating several markets makes profits surprisingly robust to spot price prediction errors.

1 Introduction

The introduction of renewable energy production into the existing power system is complicated by the inherent variability of production technologies, which harvest energy mainly from renewable sources like wind and sun. This means that it becomes increasingly challenging to maintain the real-time balance between production and consumption as the ratio of renewable energy production increases. In a Smart Grid system the inherent flexibility of consumers, such as electric vehicles, heat pumps and HVAC-systems, may be mobilized to play an active part in solving the balancing task.

To achieve this goal, however, we believe that the load control schemes must be fully responsive and non-disruptive, [1]. Consequently we investigate a setup where the actual coordinated operation of the flexible consumers is facilitated by a third party aggregator. This commercial aggregator has implemented a Virtual Power Plant, which is assumed to have direct control of a portfolio of flexible resources.

In a deregulated power market the balance between supply and demand is maintained though a series of markets operating closer and closer to the time of delivery. To make competition fair the Virtual Power Plant must enter these markets and compete on equal terms with other players such as wind farm operators and traditional power plants. The Virtual Power Plant will then help the overall goal of load balancing simply by increasing the capacity in the markets. Market mechanisms will then generate a utilization of the total available capacity, which is cheaper and more efficient.

In this paper we investigate how the Virtual Power Plant operator can potentially make a profit by trading the flexibility of electric vehicles, heat pumps and HVAC-systems in Energy and Power Markets. We first examine how the concepts of fixed and marginal costs (well known for traditional power plants, see e.g. [2] and [3]) applies to Virtual Power Plant operation. We do this in order to investigate how different comfort demands (constraint) determine how flexibility should be traded and also how much profit can be earned.

Other references, such as [4], [5], [6] have also investigated price optimized consumption scheduling for Smart Grid technologies. However, these references investigate a single-stage model where only spot price optimization is performed. In this paper we develop a three stage market model, where Day-Ahead Market, Intra-Day Market and Regulating Power Market are included. The main objective of this paper is consequently

to test the hypothesis that a Virtual Power Plant under reasonable assumptions can generate additional profit by participating in several markets.

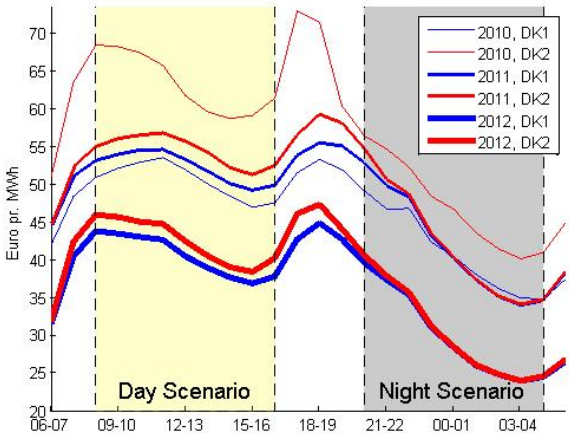

Figure 8.1: Average Spot price at each hour of the day in DK1 and DK2.

2 Fixed and Marginal Costs

In economics, fixed costs are necessary expenses, which must be covered in order to enable the production of a given product, but which are not related to the quantity or quality of product produced. In power systems, fixed costs are therefore also referred to as ”overnight” cost, because it is the present day cost of constructing a production facility ”overnight” [2]. While a Virtual Power Plant does not require the construction of a facility as such, there are a number of fixed costs, which must be covered in order for the Virtual Power Plant to be in operation. Examples of such expenses include marketing, installation, reading and maintenance of communication and metering equipment, development of IT-platform plus customer billing and accounting.

In power systems, marginal costs are defined as the costs/savings associated with producing one more/less kilowatt-hour [2]. For a power production facility, variable costs can therefore be computed as fuel cost per produced power unit plus costs of maintenance and wear. For a Virtual Power Plant, however, the calculations are more complicated. This is because most Smart Grid technologies actually do not consume one more/less kilowatt-hour, but rather advances/postpones the consumption of that kilowatt-hour.

Fixed costs should obviously be recovered over time in order for the Virtual Power Plant to prove a profitable concept. Fixed costs, however, do not affect the prices at which the Virtual Power Plant should bid into the market. For any production facility it is true that if the *market price* is higher than the *marginal cost* of production then the facility will earn (*market price* - *marginal cost*) per unit produced. Production facilities should

therefore always bid at marginal cost, since making a small profit is better than making no profit at all.

Taking this to the consumption side it is found that if the *market price* is lower than *marginal cost* of consumption then the Virtual Power Plant will earn (*marginal cost* - *market price*) per unit consumed. Consequently, the Virtual Power Plant should also bid on the market at marginal costs.

As mentioned earlier, however, a Virtual Power Plant does not simply increase or decrease its consumption, but rather advances or postpones consumption. Therefore, marginal costs of a Virtual Power Plant can only be determined if market prices are known. However, this assumption is hardly ever satisfied at the time of bidding, so the Virtual Power Plant must use a best estimate of prices to determine its own marginal costs and thus appropriate bidding price. Marginal costs for the Virtual Power Plant will be discussed and exemplified much further in Section 4 and 5.

Notice that throughout the paper, up- and down-regulation is defined in accordance with classical conventions for the consumption side. This means the up-regulation for a flexible consumer corresponds to a decrease in consumption and down-regulation corresponds to an increase in consumption.

3 Flexibility Modeling

In the present paper, flexibility is defined based on the *Buckets*, *Batteries and Bakeries*-taxonomy presented in [7]. The first model, denoted the *Bucket*, is a power and energy constrained integrator with a drain. The *Battery* is also a power and energy constrained integrator, but without the drain and with the added restriction that the unit must be fully charged at a specific deadline. Finally the *Bakery* extends the *Battery* with the additional constraint that the process must run as a batch process at constant power consumption. We let $P_i(k)$ denote the power consumption of unit i at sample k and let $E_i(k)$ denote the energy level in unit i at sample k. Note also that unless otherwise stated variables are real positive scalars.

Formal definitions of a *Bucket*, a *Battery* and a *Bakery* are given in Definition 21, 22 and 23 respectively. In the following T_s denotes the size of the time step, $\underline{P}_i$ and $\overline{P}_i$ denote limits on consumption rate, $\underline{E}_i$ and $\overline{E}_i$ denote limits on energy storage levels and $v_i(k)$ is a boolean-valued variable, which state whether a *Bakery* is running at sample k.

Definition 21 (Bucket)**.** The dynamics and constraints of a *Bucket* with drain α are

$$
\begin{aligned}
&\textit{Bucket}_i(k)\text{:}\\
&\quad E_i(k+1) = \alpha E_i(k) + T_s\big(P_i(k) + P_{i,Plan}(k)\big) && \text{(A.1)}\\
&\quad \underline{P}_i - P_{i,Plan}(k) \le P_i(k) \le \overline{P}_i - P_{i,Plan}(k) && \text{(A.2)}\\
&\quad \underline{E}_i \le E_i(k) \le \overline{E}_i && \text{(A.3)}\\
&\quad E_i(0) = E_{i,0}, && \text{(A.4)}
\end{aligned}
$$

where $k = 0, 1, \ldots, \infty$, $i = 1, 2, \ldots, N^{\textit{Buckets}}$, $0 \le \alpha \le 1$, $\underline{P}_i \le 0 \le \overline{P}_i$, $\underline{P}_i \le P_{i,Plan} \le \overline{P}_i$ and $\underline{E}_i \le E_{i,0} \le \overline{E}_i$.

Definition 22 (Battery)**.** The dynamics and constraints of a *Battery* are

$$
\begin{aligned}
&\textit{Battery}_i(k)\text{:}\\
&E_i(k+1) = E_i(k) + T_s\big(P_i(k) + P_{i,Plan}(k)\big) && \text{(B.1)}\\
&0 - P_{i,Plan}(k) \leq P_i(k) \leq \overline{P}_i - P_{i,Plan}(k) && \text{(B.2)}\\
&0 \leq E_i(k) \leq \overline{E}_i && \text{(B.3)}\\
&E_i(0) = E_{i,0}, && \text{(B.4)}\\
&E_i(T_{end,i}) = \overline{E}_i, && \text{(B.5)}
\end{aligned}
$$

where $k = 0, 1, \ldots, \infty$, $i = 1, 2, \ldots, N^{\textit{Batteries}}$, $T_{end,i} \in \mathbb{N}$, $0 \leq \overline{P}_i$, $0 \leq P_{i,Plan} \leq \overline{P}_i$ and $0 \leq \overline{E}_i$.

Definition 23 (Bakery)**.** The dynamics and constraints of a *Bakery* are

$$
\begin{aligned}
&\textit{Bakery}_i(k)\text{:}\\
&E_i(k+1) = E_i(k) + T_s\big(P_i(k) + P_{i,Plan}(k)\big), && \text{(C.1)}\\
&\overline{P}_i v_i = P_i(k) + P_{i,Plan} && \text{(C.2)}\\
&0 \leq E_i(k) \leq \overline{E}_i, && \text{(C.3)}\\
&E_i(0) = E_{i,0}, && \text{(C.4)}\\
&E_i(T_{end,i}) = \overline{E}_i, && \text{(C.5)}\\
&0 \leq \sum_{l=k}^{k+T_{run,i}-1} v_i(l) - T_{run,i}\big(v_i(k) - v_i(k-1)\big), && \text{(C.6)}
\end{aligned}
$$

where $k = 0, 1, \ldots, K$, $0 \leq \overline{P}_i$, $0 \leq P_i \leq \overline{P}_i$, $\overline{E}_i = \overline{P}_i T_{run,i} - E_{i,0}$, $v_i(k) \in \{0, 1\}$, $i = 1, 2, \ldots, N^{\textit{Bakeries}}$, $T_{end,i} \in \mathbb{N}$ and $T_{run,i} \in \mathbb{N}$.

4 Market Theory and Model

This section gives a short introduction to the Nordic Power Markets and next extends this introduction to a market model.

The Nordic Power Markets

In the Nordic countries the balance between production and consumption at the market level is maintained by means of Day-Ahead Markets, Intra-Day Markets, Regulating Power Markets and Balancing Power Markets (after-day settlement). This section gives a general description of the setup.

As the name suggests, the Day-Ahead Market (the Spot Market) operates before the actual time of delivery. Producers and wholesalers make bids for production and consumption in future time slots and prices are settled based on a double auction. Once prices on the Day-Ahead Market are settled (Market Clearing), the market is closed.

On the Day-Ahead Market producers and wholesalers have made bids based on the best available knowledge at the time of bidding. As time progresses, however, better

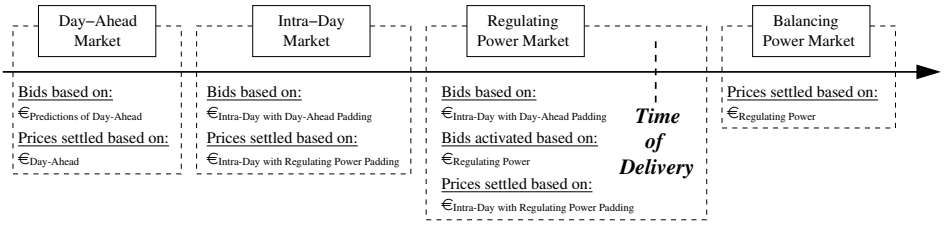

Table 8.1: Market Model.

forecasts become available. The Day-Ahead Market is therefore followed by the Intra-Day Market (the Elbas Market), where players have the option of adjusting their initial production and consumption schedules in future time slots. The Intra-Day Market is a continuous market where trading takes place up until one hour before the hour of delivery. The Intra-Day Market consists of two lists, which are continuously updated: One list for power purchases and one for power sales. Whenever there is a match within these lists (meaning that a player is willing to purchase power at a price which is higher than another players sales price), these two bids are activated and removed from the lists. This means that the Intra-Day market is more bilateral in nature than the other markets.

If players do not follow the schedule generated on the Day-Ahead and Intra-Day markets, they generate a need for balancing, i.e. up- or down-regulation. Up- and down-regulation are performed by spare capacity denoted *reserves*, which are in place because "the price mechanism cannot work fast enough to balance consumption and production in real time" [3]. Traditionally, reserves are provided by specific power plants, which are operating at less than full capacity, so they can ramp up or down as needed. In the Nordic markets reserve services are traded on the Regulating Power Market. Having a designated power market insures that a competitive price is paid for Regulating Power. In the Regulating Power Market, bids can be made up to 15 minutes before the hour of delivery. If a need for regulation arises during the hour of operation, then bids are activated in accordance with the highest price of the block of most inexpensive bids until the requested regulation is accumulated.

After the actual time of delivery, metered data of actual production/consumption is evaluated. In the after-day settlement (or Balancing Power Market), producers and wholesalers are invoiced according to their trades across the Day-Ahead, Intra-Day and Regulating Power markets. In the Balancing Power Market the cost of Regulating Power is also transferred to any player that deviated from the contracted production/consumption.

Market Model

After the introduction above, we now develop a market model based on historic data. The model consists of a series of optimization problems, which are solved one by one, each time using the latest and most updated information. Since the model is based on historic data there is no feedback in the formation of prices, meaning that prices is not generated dynamically. Consequently, the model is only valid if we assume that the amount of flexibility bid into the system is small enough not to affect the formation of the price cross significantly. On the other hand, since calculations are based only on historic data,

results are not blurred by assumptions or estimated correlations.

We denote by $(\cdot)^*_{\{t_1-t_2\}}$ a list of elements associated with each hour of the interval from t_1 to t_2, e.g. $€^*_{\{01:00-04:00\}} = [€(01:00-02:00), €(02:00-03:00), €(03:00-04:00)]$. Also $(\cdot)^*_{\{12:00-12:00\}}$ denotes values associated with a 24 hour period from 12:00 noon till 12:00 noon of the following day.

The market model has four main stages as depicted in Table 8.1 and the trading algorithm is also summarized in **Algorithm 5**.

The first stage is the Day-Ahead Market, which the Virtual Power Plant can bid into based on predictions of market prices. Day-Ahead prices are denoted $€_{\text{Day-Ahead},\{12:00\text{-}12:00\}}$, so if the Virtual Power Plant wants to maximize its profit it should bid according to the solution of

$$\min_{P(k)} \sum_{k=\{12:00-12:00\}} €_{\text{Predictions of Day-Ahead}}(k)P(k) \tag{8.1}$$

$$s.t.$$

$$(A.1)-(A.4),(B.1)-(B.5) \text{ and } (C.1)-(C.6), \tag{8.2}$$

where P_{Plan} is zero for all units and all time slots.

Based on the trading in the Day-Ahead Market a 24-hour base plan denoted $\boldsymbol{P}^*_{Plan,\{12:00-12:00\}}$ is generated.

In the next stage of the model, the Intra-Day market is opened. If there is activity on the Intra-Day market, the Virtual Power Plant can do additional trading here to further increase its profit. Often, however, there is not activity on the Intra-Day market in all hours of the day (See Figure 8.2). In an hour where there is no activity, some estimate must be used by the Virtual Power Plant to make decisions and do trading. In an hour where there is no activity on the Intra-Day market it is assumed that the Virtual Power Plant uses Day-Ahead prices as best estimation of regulating prices during that hour. These prices are denoted $€_{\text{Intra-Day with Day-Ahead Padding}}$. The Virtual Power Plant therefore bids into the Intra-Day Market according to the solution of

$$\min_{P(k)} \sum_{k=\{12:00-12:00\}} €_{\text{Intra-Day with Day-Ahead Padding}}(k)P(k) \tag{8.3}$$

$$s.t.$$

$$(A.1)-(A.4),(B.1)-(B.5) \text{ and } (C.1)-(C.6), \tag{8.4}$$

where P_{Plan} is now set according to the Day-Ahead trading.

At this time, however, the Day-Ahead Market is closed so the Virtual Power Plant cannot actually trade at spot price. Instead it must go into imbalances for which it obviously has to pay $€_{\text{Regulating Power}}$. Bidding on the Intra-Day market is therefore done based on $€_{\text{Intra-Day with Day-Ahead Padding}}$, but costs/profits are settled based on $€_{\text{Intra-Day with Regulating Power Padding}}$.

In the final stage of the market model, the Virtual Power Plant must make up- and down-regulation bids into the Regulating Power Market. In order to determine the appropriate bidding price for e.g. time slot 12:00 to 13:00 the Virtual Power Plant must

solve

$$\min_{P(k)} \sum_{k=\{13:00-12:00\}} €_{\text{Intra-Day with Day-Ahead Padding}}(k)P(k) \tag{8.5}$$

$$s.t.$$

$$(A.1)-(A.4),(B.1)-(B.5) \text{ and } (C.1)-(C.6) \tag{8.6}$$

$$P_{\textit{Bucket}}(12:00-13:00) = P_{limit,\ \textit{Bucket}} \tag{8.7}$$

$$P_{\textit{Battery}}(12:00-13:00) = P_{limit,\ \textit{Battery}} \tag{8.8}$$

$$P_{\textit{Bakery}}(12:00-13:00) = P_{limit,\ \textit{Bakery}} \tag{8.9}$$

where P_{Plan} is now the base load plan after both Day-Ahead and Intra-Day trading. Problem (8.5) to (8.9) must be solved for two values of P_{limit} in order to find both the up- and down-regulation bid price. The problem must therefore first be solved for P_{limit} equal to the maximum up-regulation adjustment that the unit can make to its consumption within the restrictions of its constraints, dynamics and base load plan between 12:00 and 13:00. Next the problem is solved for the maximum down-regulation adjustment. For each unit in the portfolio the Virtual Power Plant will then place an up- and a down-regulation bid of

$$\sum_{k=\{13:00-12:00\}} -€_{\text{Intra-Day with Day-Ahead Padding}}(k)\frac{P_{(8.5)-(8.9)}(k)}{P_{limit}} \tag{8.10}$$

where $P_{(8.5)-(8.9)}$ is the solution of (8.5)-(8.9) for each of the two values of P_{limit}.

At each remaining hour of the day the Virtual Power Plant should repeat this approach and adjust its base load plan according to $P_{(8.5)-(8.9)}$ whenever a bid is activated. Notice that up-regulation bids should be as low as possible to get activated and down-regulation bids should be as high as possible to get activated.

Finally, since Intra-Day trading and Regulating Power trading are settled at $€_{\text{Intra-Day with Regulating Power Padding}}$ we do not need an independent stage for the balancing market, since the appropriate imbalances have already been paid/compensated at the price of Regulating Power.

Algorithm 5: Market Trading

from January 1^{st} **to** December 31^{st}
  Generate €$_{\text{Predictions of Day-Ahead}}$ and solve (8.1) to (8.2).
  Purchase $\boldsymbol{P}^*_{Plan}$ in the Day-Ahead Market according to solution of (8.1) to (8.2).
  Retrieve €$_{\text{Day-Ahead}}$ and €$_{\text{Intra-Day}}$ and generate €$_{\text{Intra-Day with Day-Ahead Padding}}$.
  Solve (8.3) to (8.4) and update $\boldsymbol{P}^*_{Plan}$ according to purchase/sales in the Intra-Day market.
  from 12:00 **to** 11:00
    Solve (8.5) to (8.9) and bid into the Regulating Power
    Market according to (8.10).
    If bid activated **then** update $\boldsymbol{P}^*_{Plan}$.
  end from
end from

5 Analysis and Results

Market Data

The present analysis focuses on the Danish Power Market, so Day-Ahead prices, average Intra-Day prices and Regulating Power prices for DK1 (Western Denmark) and DK2 (Eastern Denmark) in 2010, 2011 and 2012 form the basis of the main analysis. The data set can be downloaded from [8]. Figure 8.1 shows the average Day-Ahead price for each hour of the day. In later simulations we will consider a day and a night scenario (inspired by an electric vehicle in frequent use). These scenarios are also depicted in Figure 8.1.

Single-Day Illustration

Based on the taxonomy presented in Section 3 and the market model developed in Section 4 we now illustrate how the Virtual Power Plant should bid each of the flexibility types in the taxonomy into the markets. The algorithm is based on the assumption that the Virtual Power Plant is continuously trying to maximize its profit based on the best available knowledge. Prices from February 3^{rd}, 2012 in DK1 from 08:00 to 17:00 are randomly chosen as illustrating (see Figure 8.2). We consider a portfolio consisting of one unit of each type in the taxonomy, with parameters values $\alpha = 0.9, \underline{P} = -1MW, \overline{P} = 1MW, \underline{E} = 0MWh, \overline{E} = 3MWh$ for all units, which corresponds to $T_{run} = 3\ hours$ for the *Bakery*.

The first prices that are settled are the Day-Ahead Prices. If we assume that the Virtual Power Plant is capable of predicting these prices exactly (the implications of this assumption will be investigated further in Section 5), then the Day-Ahead base load plan for each of the units will be as in Figure 8.3. Here it can be seen that the *Battery* and *Bakery* have paid 165 € to satisfy their base load requirements and that the *Bucket* has not yet made any profit.

Next the Intra-Day Market is opened. It can be seen from Figure 8.2 that in DK1 there was only trading in the Intra-Day Market between 14:00 and 17:00 on February 3^{rd}, 2012

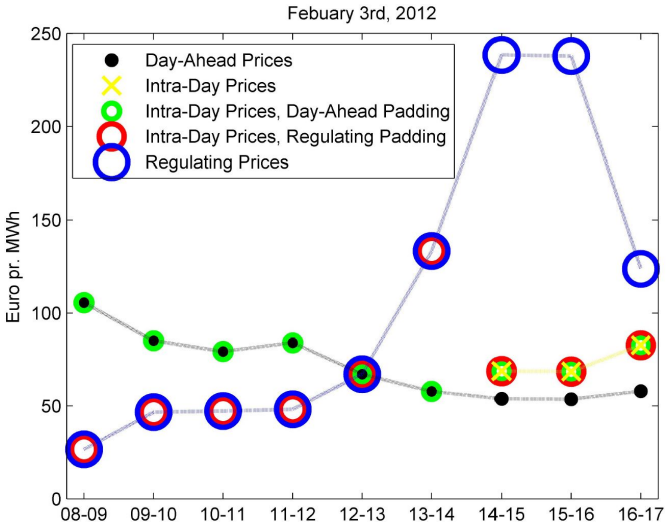

Figure 8.2: Electricity Prices in DK1 on February 3^{rd}, 2012.

(yellow crosses). Since there is some activity on the Intra-Day market, the Virtual Power Plant can try to make an additional profit on new trading. The results of Intra-Day trading are shown Figure 8.3. The *Battery* and *Bakery* sell power between 14:00 and 15:00 and between 15:00 and 16:00 respectively. They are then scheduled to over consume between 12:00 and 13:00 instead, because spot prices were low here. This is a gamble, because the Day-Ahead Market is no longer open and there is no activity on the Intra-Day market in the time slot between 12:00 and 13:00, so the Virtual Power Plant-Operator cannot buy the power anywhere. However, since the regulating price between 12:00 and 13:00 ends up equal to the spot price the gamble earns 1 € per unit.

In the Intra-Day market the *Bucket* is scheduled to overconsume 1 MWh between 15:00 and 16:00 in order to be able to sell 0.9 MWh between 16:00 and 17:00. This earns the *Bucket* a profit of 6 €. Unfortunately it is scheduled to do the same between 13:00 and 15:00. If the regulating power price at time 13:00-14:00 had equaled the spot price, then this would have earned the *Bucket* a profit of 4 €. As it turns out, however, the regulating price at 13:00-14:00 is 133 €, so the trade would actually costs the *Bucket* 72 €, if it had done no further trading on the considered day. Later in the day, however, the *Bucket* bids into the Regulating Power Market during that critical hour and therefore the trading will not be as costly to the *Bucket* as it first looked. This is exactly the advantage of trading in several markets.

Next the Virtual Power Plant must bid into the Regulating Power Market for the time slot 08:00 to 09:00. Since no power have been purchased for any units to consume during this time slot, the Virtual Power Plant cannot make any up-regulation bids (recall that up and down regulation is defined based on the production conventions, so up-regulation corresponds to a decrease in consumption). It can, however, make three down-regulation bids:

- The *Bucket* is bid at 77 € for 1 MW down-regulation. If the *Bucket* gets activated

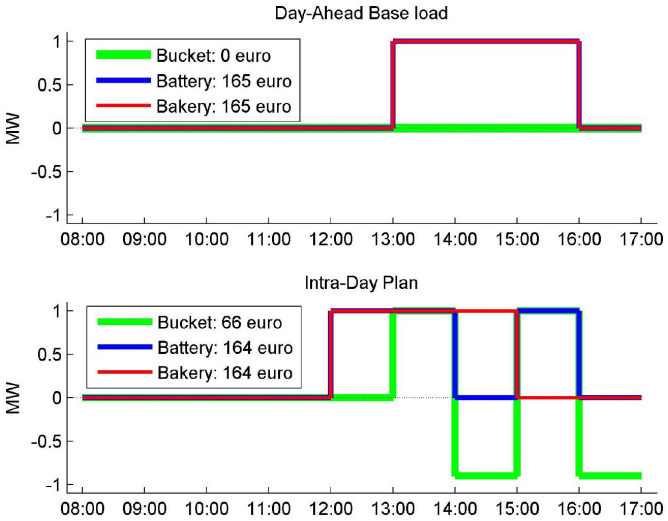

Figure 8.3: Base load plan based on perfect prediction of Day-Ahead prices and Intra-Hour plan based on trading in open Day-Ahead time slots and the assumption that Regulating Power prices will equal Spot prices in DK1 on February 3^{rd}, 2012 for each type of flexibility in the taxonomy.

for down regulation (increase in consumption), it will be able to under-consume 0.9 MWh between 09:00 and 10:00. Given the current expected prices, the Virtual Power Plant assumes that this will save 77 €, so if up-regulation power can be purchased for less than 77 € between 08:00 to 09:00, a profit will be made.

- The *Battery* is bid at 68 € for 1 MW down-regulation. If the *Battery* gets activated for down-regulation between 08:00 to 09:00, then it will be able to under consume between 15:00 and 16:00, which is expected to save 68 €. So again, if up-regulation power can be purchased for the *Battery* at less than 68 €, then a profit will be made.

- The *Bakery* is bid at 29 € for 1 MW down-regulation. If the *Bakery* should start early, then it must sell all power between 12:00 and 15:00 and over-consume between 09:00 and 11:00. This change to the consumption schedule is expected to cost 29 €, so down-regulation power at time 08:00-09:00 must be cheaper than 29 € in order for the *Bakery* to be interested in moving its consumption.

The down-regulation power price between 08:00 and 09:00 comes out at 26 €, so all bids are activated. Now the cost associated with the *Bucket* drops down to 50 € and the *Battery* has paid 122 € for its total power consumption, if it does no more trading today. The *Bakery* is very lucky, as it turns out that by starting earlier it will end up paying just 13 € for its total power consumption that day.

Table 8.2 states all the bids that the Virtual Power Plant would make on February 3^{rd}, 2012, if it continuously attempts to maximize its profit based on the best available knowledge. The adjusted consumption plans for the units as the day progresses are depicted in Figure 8.4. By the end of the day, the *Bucket* has earned 199 € while both the *Battery*

and the *Bakery* have paid only 15 € for their total power consumption compared to their initial cost of 165 €. Thus offering to be flexible on February 3^{rd}, 2012 could have turned out to be very good business.

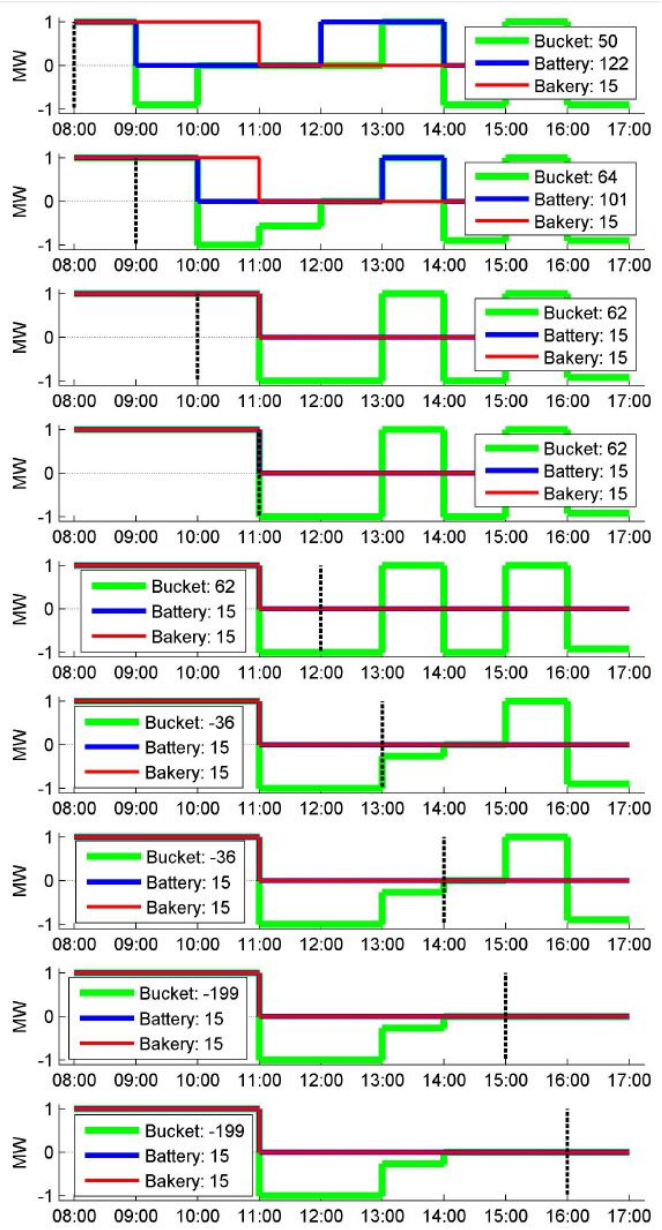

Figure 8.4: Consumption plan for each hour of February 3^{rd}, 2012, DK1 as trading in the Regulating Power Market progresses.

		Bucket		*Battery*		*Bakery*	
	Activated	**Down**	**Up**	**Down**	**Up**	**Down**	**Up**
08-09	Down	77	N/A	68	N/A	29	N/A
09-10	Down	67	N/A	67	N/A	N/A	N/A
10-11	Down	56	N/A	58	N/A	N/A	N/A
11-12	Down	44	N/A	N/A	N/A	N/A	N/A
12-13		52	N/A	N/A	N/A	N/A	N/A
13-14	Up	N/A	56	N/A	N/A	N/A	N/A
14-15	Up	62	N/A	N/A	N/A	N/A	N/A
15-16	Up	N/A	74	N/A	N/A	N/A	N/A
16-17	Up	0	N/A	N/A	N/A	N/A	N/A

Table 8.2: Bids made into the Regulating Power Market. Entries are underlined if bids are available/activated.

Notice especially that when the *Bucket* did trading on the Intra-Day Market, its total cost went up, because it did not know that the Regulating Power prices would be unfavourable later. By bidding into the Regulating Power Market, however, the *Bucket* saves itself from actually consuming between 13:00 and 14:00 where regulating prices are high and therefore recovers its loss.

Full Year Simulations

In the next simulation example, we consider one unit of each type in the taxonomy. Parameter values are again $\alpha = 0.9, \underline{P} = -1MW, \overline{P} = 1MW, \underline{E} = 0MWh, \overline{E} = 3MWh$. The *Battery* and the *Bakery* are limited to trading between 08:00 to 17:00 (Day) and 20:00 to 05:00 (Night), whereas the *Bucket* is allowed to trade round-the-clock.

The results of trading the portfolio according to the algorithm given in Section 4 are given in Table 8.3 and 8.4. It can be seen that profits/savings do indeed increase as more markets are penetrated. The benefit is largest for the *Bucket* and relatively limited for the *Bakery*. However, it is on average always beneficial to offer flexibility to the system. In some scenarios, such as the *Battery*, 2012, DK1, Day-scenario costs actually increase from Day-Ahead to Intra-Day market. This is because Day-Ahead is done with perfect prediction of prices, whereas Intra-Day trading is sometimes settled at Regulating Power price. After the *Battery* has participated in the Regulating Power Market there are still savings to be obtained by trading in several markets, however.

Virtual Power Plant Profit

Since the *Battery* and *Bakery* both have base load requirements to satisfy, it is possible for the Virtual Power Plant to achieve savings, but not an actual profit as is the case for the *Bucket*. A sensible agreement between the unit owner and the Virtual Power Plant could thus be that the unit owner should cover the Day-Ahead base load cost. Any additional profit gained in the Intra-Day and Regulating Power Markets should then be shared evenly between the unit owner and the Virtual Power Plant. With this setup, Table 8.5 shows

Year	Area	***Bucket*** Day-Ahead	Intra-Day	Regulating
2010	DK1	-6.823	-7.544	-20.792
	DK2	-21.409	-21.998	-28.937
2011	DK1	-8.655	-9.969	-27.880
	DK2	-10.156	-11.386	-30.047
2012	DK1	-11.707	-14.639	-35.981
	DK2	-14.541	-17.383	-39.282
Total		-73.292	-82.919	-182.919

Table 8.3: Profit/savings in € obtained by trading the *Bucket* according to the algorithm given in Section 4.

Year	Area	Scenario	***Battery*** Day-Ahead	Intra-Day	Regulating	***Bakery*** Day-Ahead	Intra-Day	Regulating
2010	DK1	Day,	50.984	51.281	48.877	51.307	51.432	50.793
		Night	38.154	38.047	35.083	38.233	38.095	35.605
	DK2	Day	62.139	62.367	58.965	62.453	62.872	61.609
		Night	44.067	43.927	42.775	44.170	43.935	43.329
2011	DK1	Day,	53.178	52.488	47.403	53.404	52.440	50.425
		Night	37.376	36.873	34.103	37.463	37.041	35.686
	DK2	Day	55.192	55.001	47.221	55.460	55.786	51.084
		Night	37.539	37.449	35.430	37.621	37.403	36.638
2012	DK1	Day,	39.843	39.345	31.409	40.059	40.058	37.176
		Night	26.638	26.296	24.587	26.682	26.531	25.895
	DK2	Day	41.171	40.186	31.188	41.420	40.757	37.393
		Night	26.724	26.317	24.716	26.752	26.449	26.048
Total			513.005	509.576	461.757	515.025	512.799	491.681

Table 8.4: Profit/savings in € obtained by trading the *Battery* and *Bakery* according to the algorithm given in Section 4.

which flexibility type is most profitable for the Virtual Power Plant. Is is found that the *Bucket* is far more profitable than the *Battery*, which again generates more than twice as much profit as the *Bakery*.

Sensitivity Analysis

Since we have assumed perfect prediction of Day-Ahead prices and since the *Battery* and *Bakery* savings are relatively limited it is relevant to investigate how sensitive the savings are to prediction errors on Day-Ahead prices. To do this all calculations are repeated, but now units are not allowed to purchase power during the cheapest hour in the Day Ahead market. The results are given in Table 8.6 and it is found, that the savings are surprisingly unaffected by the prediction error: Just 0.5% and 1.1% increases in costs for the *Battery* and the *Bakery*, respectively.

It has also been investigated how the *Bucket* profit is affected by the size of the energy drain. Again parameter values are $\alpha = 0.9, \underline{P} = -1MW, \overline{P} = 1MW, \underline{E} = 0MWh, \overline{E} = 3MWh$ and the results are depicted in Figure 8.5. As expected the profit is

Year	Area	***Bucket***	Scenario	***Battery***	***Bakery***
2010	DK1	-10.396	Day	-1.053	-257
			Night	-1.535	-1.314
	DK2	-14.469	Day	-1.587	-422
			Night	-646	-420
2011	DK1	-13.940	Day	-2.888	-1.489
			Night	-1.636	-889
	DK2	-15.023	Day	-3.986	-2.188
			Night	-1.054	-492
2012	DK1	-17.991	Day	-4.217	-1.442
			Night	-1.025	-393
	DK2	-19.641	Day	-4.991	-2.014
			Night	-1.004	-352
Total		-91.459		-25.624	-11.672
Percentage		100%		14%	6%

Table 8.5: Virtual Power Plant profit in €, when the Day-Ahead base load costs of the *Battery* and *Bakery* are covered by the unit owner.

	Battery	***Bakery***
Perfect Prediction	461.757	491.681
With error	464.064	497.147
Difference	2.308	5.466
Percentage	0.5%	1.1%

Table 8.6: Increase in costs when the *Battery* and *Bakery* are not allowed to purchase power during the cheapest hour on the Day-Ahead Market.

heavily influenced by the size of the energy drain, but even with a drain of 20% per hour the total profit is still more than 50.000 €.

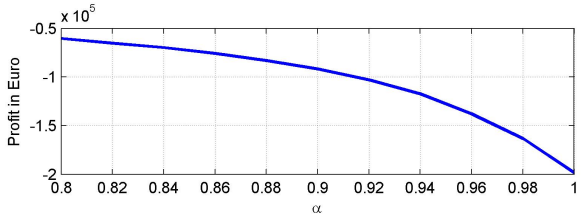

Figure 8.5: Profit of the *Bucket* summarized over DK1 and DK2 and 2010, 2011 and 2012 as a function of energy drain.

6 Conclusion

In this paper we have developed a three stage electric power market model, which include Day-Ahead (Spot), Intra-Day and Regulating Power Markets. By use of this model we have confirmed the hypothesis that a Virtual Power Plant operator can increase its profit by trading in several markets from day to day. We have also found that the profit is highly sensitive to the type of flexibility considered, but surprisingly robust to errors in Day-Ahead price predictions.

References

[1] Duncan S. Callaway and Ian A. Hiskens, *Achieving Controllability of Electric Loads*, Proceedings of the IEEE, Vol. 99, No. 1, January 2011.

[2] S. Stoft, *Power System Economics Designing Markets for Electricity*, Wiley-IEEE Press 1 (1) .

[3] I. Wangensteen, *Power System Economics - the Nordic Electricity Market*, Tapir Academic Press, 2007.

[4] Kin Cheong Sou, James Weimer, Henrik Sandberg, and Karl Henrik Johansson, *Scheduling Smart Home Appliances Using Mixed Integer Linear Programming*, 50th IEEE Conference on Decision and Control and European Control Conference, 2011.

[5] Matt Kraning, Yang Wang, Ekine Akuiyibo, Stephen Boyd, *Operation and Configuration of a Storage Portfolio via Convex Optimization*, 18th IFAC World Congress, 2011, pp. 10487-10492.

[6] Nikolaos Gatsis and Georgios B. Giannakis, *Residential Demand Response with Interruptible Tasks: Duality and Algorithms*, 50th IEEE Conference on Decision and Control and European Control Conference, 2011, pp. 1-6.

[7] M. K. Petersen, K. Edlund, L. H. Hansen, J. Bendtsen and J. Stoustrup, *A Taxonomy for Flexibility Modeling and a Computationally Efficient Algorithm for Dispatch in Smart Grids*, American Control Conference, 2013.

[8] `http://www.energinet.dk/EN/El/Engrosmarked/Udtraek-af-markedsdata/Sider/default.aspx`

Paper E

Heuristic Optimization for the Discrete Virtual Power Plant Dispatch Problem

Mette Kirschmeyer Petersen, Lars Henrik Hansen, Jan Dimon Bendtsen, Kristian Edlund and Jakob Stoustrup

This paper has been published in IEEE Transaction on Smart Grid, Vol. 5, No. 6, 2014.

Abstract

We consider a Virtual Power Plant, which is given the task of dispatching a fluctuating power supply to a portfolio of flexible consumers. The flexible consumers are modelled as discrete batch processes, and the associated optimization problem is denoted the Discrete Virtual Power Plant Dispatch Problem.

First NP-completeness of the Discrete Virtual Power Plant Dispatch Problem is proved formally. We then proceed to develop tailored versions of the meta-heuristic algorithms **Hill Climber** and **Greedy Randomized Adaptive Search Procedure** (**GRASP**). The algorithms are tuned and tested on portfolios of varying sizes.

We find that all the tailored algorithms perform satisfactorily in the sense that they are able to find sub-optimal, but usable, solutions to very large problems (on the order of 10^5 units) at computation times on the scale of just 10 seconds, which is far beyond the capabilities of the optimal algorithms we have tested. In particular, GRASP Sorted shows the most promising performance, as it is able to find solutions that are both agile (sorted) and well balanced, and consistently yields the best numerical performance among the developed algorithms.

1 Introduction

Global efforts to reduce CO_2 emissions has driven the introduction or renewable power generation technologies into the power system. However, since solar panels and wind turbines harvest energy from sun and wind power availability becomes changeable and more difficult to predict. The Smart Grid was born out of the need to maintain the balance between production and consumption in this far more volatile power system. In the Smart Grid a communication link to the consumption side is established, such that flexible consumers like electric vehicles, heat pumps and HVAC-systems can be organized and activated to follow power availability and scarcity, see [1] and [2].

A major challenge in developing the Smart Grid is the sheer size of the optimization problems involved. Solving a dispatch problem for a traditional power system with tens or hundreds of generators is a challenge, which has been researched for decades, see [3], [4], [5]. Switching to the Smart Grid, however, will expand that problem with additional thousands or millions of units. This means that the computation time associated with determining an optimal solution is very likely to be unacceptable in practice.

A review of computation times for optimization in Smart Grid literature reveals that computation time is still a very real challenge. Table 9.1 summarizes problem size and computation times for ten recent scientific publications related to Smart Grid optimization. Obviously computation times are highly dependent on the specific structure of the considered problem and the software and platform used for calculation. Nonetheless, Table 9.1 does give the general impression that computation times are still quite a lot longer than what one can expect to be acceptable for a fully deployed Smart Grid operating in real time. This is the case even though several of the cited references investigate heuristic rather than exact optimization methods.

The fastest results found are given in [8] with computation times of just 1.2 seconds. In [8], however, it is shown that the considered method does not scale to larger problems due to memory overload.

To substantiate the aforementioned assertion, the present paper introduces the Discrete Virtual Power Plant Dispatch Problem in which batch processes (constant power

Reference	Simulation samples	Num. units	Comp. time
[8]	144	<20	1.2 sec
[9]	24	600	30 sec
[10]	24	5	50 sec
[11]	Unclear	20858	1.1 min
[12]	288	6	3.7 hours
[13]	24	3504	6 hours
[14]	24	50	83 hours
[15]	12	3	Not stated
[16]	144	100	Not stated

Table 9.1: Overview of computation times reported in recent scientific publications on optimization in Smart Grid applications. Simulation samples is the number of discrete time steps that the considered simulation horizon has been split into.

consumption with fixed duration) must be scheduled to balance a fluctuating power supply. Similar optimization problems have been investigated in [6], [7], [8] and [16]. In these papers batch processes are used as simple models of electric vehicles, dishwashers, microwave ovens and more. These papers formulate objectives to schedule units to balance a fluctuating, limited or costly power supply.

After formulating the Discrete Virtual Power Plant Dispatch Problem it is proven formally that the optimization problem is NP-complete [17]. Motivated by this knowledge, we proceed to investigate the performance of two heuristic methods to obtain solutions, which are sub-optimal, but fast to compute. This way a feasible solution will always be available before some predefined Power Market Gate Closure; In practice, a suboptimal solution available two minutes before market gate closure is far more valuable than an optimal one available two minutes after.

The methods we will investigate are known in the literature as *Hill Climber* and *GRASP*. The main reason for choosing these methods to solve the Discrete Virtual Power Plant Dispatch Problem is that the considered cost function, can be implemented using Delta Evaluation; and both *Hill Climber* and *GRASP* are able to exploit that.

Delta Evaluation means that if the cost associated with on candidate solution is known, and a new solution is obtained through a limited number of permutations of the old solution, then the cost function for the new solution can be computed only from the permutations. Using delta evaluation can only improve calculation time by a constant factor; however, this factor is usually so large, that the actual improvement is far better than algorithmic changes.

Other methods that could be considered for solving the Discrete Virtual Power Plant Dispatch Problem are Ant Swarm optimization [11], Genetic Algorithms [19] and tabu search [22]. However, these algorithms are all based on manipulating a set of candidate or tabu solutions. They thus have a tendency to drown in logistics as the majority of the available computation time is spent running through and updating solution sets.

The main contributions of the paper is firstly the proof that the Discrete Virtual Power Plant Dispatch Problem problem is NP-complete. Secondly, we also show that even though finding optimal solutions of the considered problem is indeed very challenging,

highly promising results can be obtained in short time frames and for very large problems by special adaptations of the considered heuristic algorithms.

The paper is structured as follows: Firstly Section 2 presents the Discrete Virtual Power Plant Dispatch Problem including flexibility modeling and agility. In Section 3 the computational complexity of the Discrete Virtual Power Plant Dispatch Problem is explored in several different ways: Firstly, the NP-completeness is proven formally and then the feasibility of solving the problem by use of the optimization package CPLEX [18] is explored. In Section 4 four heuristic algorithms are developed, namely *Uniform Selection Hill Climber*, *Weighted Selection Hill Climber*, *GRASP Random* and *GRASP Sorted*. These algorithms are tuned and compared in Section 5. Finally, Section 6 summarizes general conclusions and suggestions for future work.

2 Optimization Problem

We consider a Virtual Power Plant, which is given the task of satisfying the consumption needs of a portfolio of flexible systems (distributed energy resources) by dispatching a fluctuating power supply.

A forecast of the fluctuating power supply is denoted $P_{Dispatch}(k), k = 1, 2, \ldots, K$, and the flexible consumers are denoted local units (LUs). A portfolio of N local units is denoted $\{LU_i\}_{i=1,2,\ldots,N}$. At sample k we let $P_i(k)$ denote the power to be dispatched to local unit i, and any quantity, which cannot be dispatched to the portfolio, is denoted $\mathcal{S}(\cdot)$. The objective is to minimize the residual power, that is $|\mathcal{S}(\cdot)|$.

The optimization problem can be formulated as

$$\min_{P_i(\cdot)} \sum_{k=1}^{K} w_k |\mathcal{S}(k)| \tag{9.1}$$

$$s.t.$$

$$P_{Dispatch}(k) \in \mathbb{R}_+, k = 1, 2, ..., K \tag{9.2}$$

$$\sum_{i=1}^{N} P_i(k) + \mathcal{S}(k) = P_{Dispatch}(k), \tag{9.3}$$

and also subject to the dynamics and constraints of $\{LU_i\}_{i=1,2,\ldots,N}$. Here K denotes the total simulation horizon.

The impatience weights $w_k \in \mathbb{R}$ have been added, because the forecasted power production will rarely fit exactly with the power needed to satisfy $\{LU_i\}_{i=1,2,\ldots,N}$. In practice, when this happens, a traditional power plant will have to adjust its power consumption, such that the discrepancy between supply and demand is compensated. As a rule of thumb, however, the better time the plant operator has to modify the output of a traditional power plant, the cheaper it can be done. It is therefore desirable to introduce slack as late on the simulation horizon as possible, which can be achieved by introducing $w_k, k = 1, 2, \ldots, K$ and requiring that $w_{k_1} > w_{k_2}$ if $k_1 < k_2$.

Flexibility Modeling

In this paper the flexible units are modeled as batch-processes, which are characterized by constant power consumption, $\overline{P}$, a run time, K_{Run}, and a deadline, K_{End}, by which

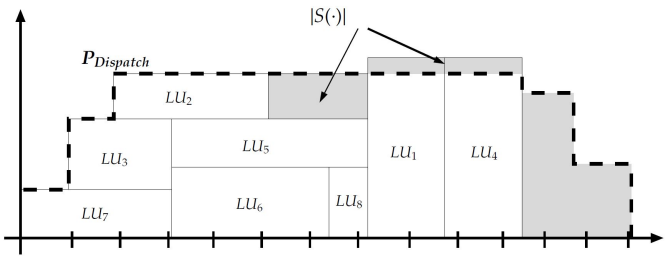

Figure 9.1: Solving the Discrete Virtual Power Plant Dispatch Problem essentially corresponds to packing rectangles under an arbitrary profile.

the process must be finished. Also we let T_s denote the size of the time step. Each local unit can therefore be modeled by

$$E(k+1) = E(k) + T_s P(k), \tag{9.4}$$

$$P(k) = \overline{P} v(k) \tag{9.5}$$

$$0 \leq E(k+1) \leq \overline{E}, \tag{9.6}$$

$$E(K_{End}) = \overline{E}, \tag{9.7}$$

$$0 \leq \sum_{l=k}^{k+K_{Run}-1} v(l) - K_{Run}\Big(v(k) - v(k-1)\Big). \tag{9.8}$$

where $\overline{P} \in \mathbb{R}_+$, $k = 1, 2, \ldots, K$, $v(k) \in \{0, 1\}$, $\overline{E} = \overline{P} K_{Run}$, $K_{Run} \leq K$ and $K_{End} \leq K$. Inequality (9.8) is the minimum runtime constraint, which ensures that if $v(k) - v(k-1)$ is one, then $v(l), l = k+1, k+2, \ldots, K_{Run} - 1$ must also all be one; that is, once the local units is activated, it must complete its consumption immediately.

We let $\{LU_i\}_{i=1,2,\ldots,N}$ denote a portfolio of N flexible consumers. Then, for each $\{LU_i\}_{i=1,2,\ldots,N}$ a solution of problem (9.1)-(9.3) consists of a set of start times, $\boldsymbol{K_{Start}} = (K_{Start,1}, K_{Start,2}, \ldots, K_{Start,N})$, one for each local unit. An illustration of the Discrete Virtual Power Plant Dispatch Problem for batch processes is given in Figure 9.1.

Agility

As mentioned earlier, $P_{Dispatch}$ is a forecast of the power production of some renewable production technology. This means that $P_{Dispatch}$ will not correspond exactly to the power, which will actually be produced over the considered time horizon. To alleviate this issue, we want to find a solution of problem (9.1)-(9.3), which is as *agile* as possible. Finding an agile solution means prioritizing the most urgent tasks, namely the ones which are closest to their deadline. In this way we will have created more maneuverability, if updated forecasts of $P_{Dispatch}$ give very different power availability than originally projected. We also call this maximizing the agility of the portfolio (see Figure 9.2 and 9.3).

The concept of agility is illustrated in Figure 9.2 and 9.3 for a portfolio consisting of three units. The top illustrations in Figure 9.2 and 9.3 both depict optimal solutions of this instance of the Discrete Virtual Power Plant Dispatch Problem and if the predicted

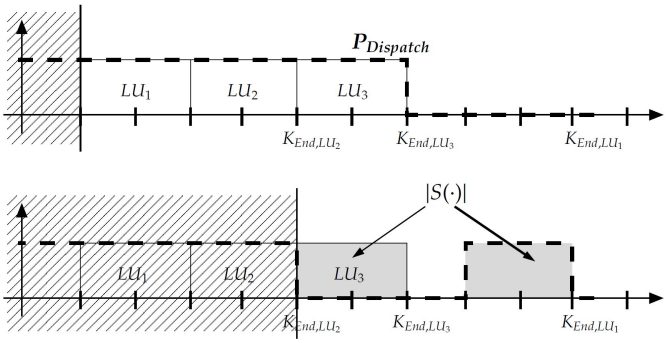

Figure 9.2: As time progresses an un-agile solution can become sub-optimal when earlier projections of $P_{Dispatch}$ turn out to be erroneous.

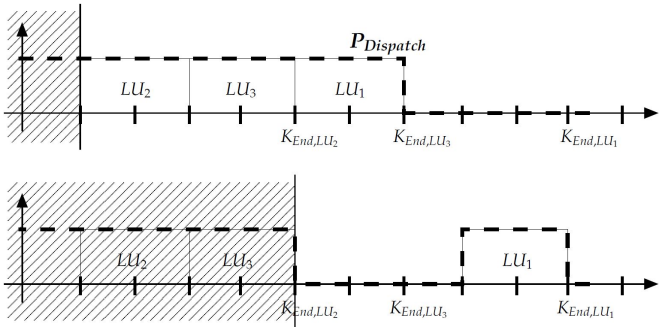

Figure 9.3: As time progresses an agile solution is more likely to remain optimal even when earlier projections of $P_{Dispatch}$ were erroneous.

progress of $P_{Dispatch}$ is correct, then it is obviously unimportant to distinguish between the two. If, however, a significant portion of the available power is delayed as in the lower illustrations, then the top solution in Figure 9.3 remains optimal, but the top solution in Figure 9.2 does not. This happens because the top solution in Figure 9.2 has left more maneuverability for the optimization in later time steps. Thus, in a sense introducing agility to the problem solving is an attempt to maximize the flexibility of the remaining solution space.

To find an agile solution of problem (9.1)-(9.3) the cost function is extended with a

term, which adds a penalty to dispatching each local unit, that is

$$f(P(\cdot)) = \min_{P_i(\cdot)} \sum_{k=1}^{K} \Big(w_k |\mathcal{S}(k)| + \sum_{i=1}^{N} w_{i,k} |P_i(k)| \Big) \tag{9.9}$$

$$s.t.$$

$$P_{Dispatch}(k) \in \mathbb{R}_+, k = 1, 2, ..., K \tag{9.10}$$

$$\sum_{i=1}^{N} P_i(k) + \mathcal{S}(k) = P_{Dispatch}(k). \tag{9.11}$$

The agility weights $w_{i,k}$ should then be chosen such that local unit i is dispatched before local unit j, if $K_{End,i} - K_{Run,i} < K_{End,j} - K_{Run,j}$.

To explain how agility weights are chosen, let $\{LU_i\}_{i=1,2,...,N}$ denote a set of local units *sorted according to deadline minus run time*. Intuitively, agility weights will penalize increasing the energy term of each local unit and this penalty is proportional to the unit index. This means replacing the cost function (9.9) with

$$\min_{P_i(\cdot)} \sum_{k=1}^{K} \Big(w_k |\mathcal{S}(k)| + \sum_{i=1}^{N} i |E_i(k)| \Big). \tag{9.12}$$

However, since $E_i(k) = \sum_{l=1}^{k} T_s P_i(l)$ Equation (9.12) can be written as

$$\min_{P_i(\cdot)} \sum_{k=1}^{K} \Big(w_k |\mathcal{S}(k)| + \sum_{i=1}^{N} i(K+1-k) T_s |P_i(k)| \Big)$$

and the agility weights are therefore given by

$$w_{i,k} = i(K+1-k)T_s. \tag{9.13}$$

Portfolio Generation for Simulation

Thoughout this paper we consider optimization problems on randomly generated portfolios. Each portfolio is characterized by the numbers N and K, such that $Portfolio(N, K)$ denotes a randomly generated portfolio of N local units with $K_{Run} \in \{2, 3, 4, 5\}$, $\overline{P} \in \{1, 2, 3, 4\}$ and $K_{End} \in \{1, 2, \ldots, K\}$. Also we set $T_s = 1$ in all simulations and all calculations have been performed on a standard laptop.

Figure 9.4 depicts a solution of problem (9.9) to (9.11) for a $Portfolio(10^5, 100)$ computed by use of the algorithm **GRASP Random**, which is introduced in Section 4. It can be seen that the majority of slack is introduced towards the end of the simulation horizon as the total consumption (blue) is no longer able to follow $P_{Dispatch}$ (black). Agility/sortedness is illustrated by green, red, cyan and magenta lines illustrating the accumulated consumption of the first, second, third and fourth quarter of units, respectively, when the portfolio is sorted according to deadline minus runtime.

3 Computational Complexity

In this section we will investigate the computational complexity of the Discrete Virtual Power Plant Dispatch Problem. We first do this by proving that the problem is NP-

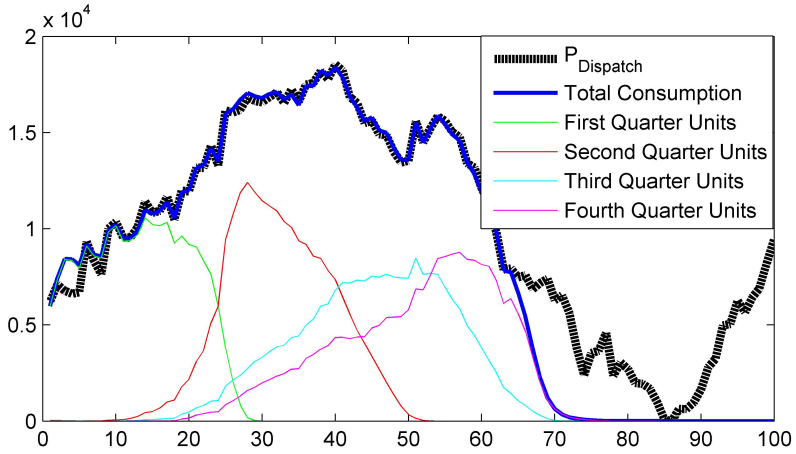

Figure 9.4: Solution of problem instance of the Discrete Virtual Power Plant Dispatch Problem with 10^5 local units computed by use of the algorithm **GRASP Random** introduced later in the paper.

Complete. Next we attempt to solve the optimization problem via CPLEX. All calculations are performed on a standard laptop.

Proof of NP-Completeness

Polynomial-time reductions provide a formal means of showing that one problem is at least as hard to solve as another to within a polynomial-time factor. That is, if $L_1 \leq_P L_2$, then L_1 is not more than a polynomial factor harder than L_2, [17].

Definition 24 (Subset-Sum Problem)**.** Let there be given a finite set $S \in \mathbb{N}$ and a target $t \in \mathbb{N}$. Is there a subset $S' \in S$ whose elements sum to t?

Lemma 6. *The Subset-Sum Problem is NP-complete.*

Proof. The proof of NP-Completeness of the Subset-Sum problem is done by formulating the $3 - CNF - SAT$ (3-conjuncture-normal-form-satisfiability) language. Next it is proved that satisfiability of boolean formulas in $3-CNF-SAT$ is NP-complete. Finally the Subset-Sum problem is formulated as boolean formulas in $3 - CNF - SAT$, thus proving that $L_{3-CNF-SAT} \leq_P L_{Subsetsum}$. For full proof see [17]. □

In the following, we show that a simplified version of the Discrete Virtual Power Plant Dispatch Problem is equivalent to the Subset-Sum problem. We shall refer to this reduced problem as DVPPDP for brevity.

Theorem 7. *The DVPPDP is NP-complete.*

Proof. In this proof it is shown that a subset of the class of instances of the Discrete Virtual Power Plant Dispatch Problem is equivalent to the Subset-Sum Problem. By poly-

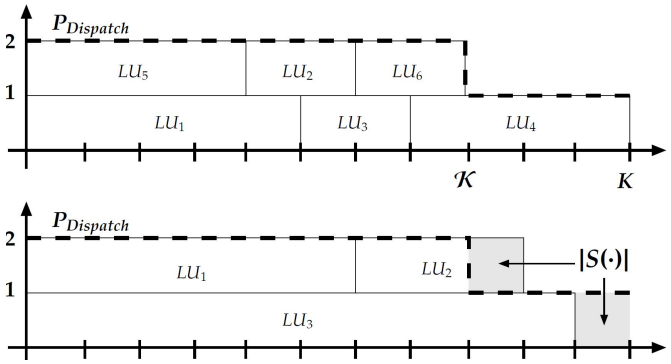

Figure 9.5: For the considered instance of the DVPPDP a solution of problem (9.14) - (9.16) such that $f(P_i(\cdot)) = 0$ can exist if and only if there also exists a subset of $K_{Run,i}, i = 1, 2, \ldots, N$, which sums to $\mathcal{K}$.

nomial reduction this proves that the Discrete Virtual Power Plant Dispatch Problem is NP-complete, since the Subset-Sum Problem is NP-complete.

Firstly we simplify the Discrete Virtual Power Plant Dispatch Problem by setting agility weights to 0. This reduces problem (9.9) - (9.11) to

$$f(P_i(\cdot)) = \min_{P_i(\cdot)} \sum_{k=1}^{K} w_i |\mathcal{S}(k)| \tag{9.14}$$

$s.t.$

$$P_{Dispatch}(k) \in \mathbb{R}_+, k = 1, 2, ..., K \tag{9.15}$$

$$\sum_{i=1}^{N} P_i(k) + \mathcal{S}(k) = P_{Dispatch}(k). \tag{9.16}$$

Flexible units are still modeled by (9.4)-(9.8).

Let $\mathcal{K} \in \mathbb{N}_+$ and $K \in \mathbb{N}_+$ be given and assume without loss of generality that $\mathcal{K} < K$. Next define portfolio $\{LU_i\}_{i=1,2,\ldots,N}$, such that $\overline{P}_i = 1, i = 1, 2, \ldots, N$, $K_{End,i} = K, i = 1, 2, \ldots, N$ and $\sum_{i=1}^{N} K_{Run,i} = \mathcal{K} + K$. Also define $P_{Dispatch}(k) = 2, k = 1, 2, \ldots, \mathcal{K}$ and $P_{Dispatch}(k) = 1, k = \mathcal{K} + 1, \mathcal{K} + 2, \ldots, K$ (see Figure 9.5).

To prove NP-completeness we formulate the following decision problem: Given the DVPPDP instance constructed above does there exist a solution of problem (9.14) - (9.16) for which $f(P_i(\cdot)) = 0$?

First observe that $\sum_{k=1}^{K} P_{Dispatch}(k) = \mathcal{K}+K$ and $\sum_{i=1}^{N} K_{Run,i}\overline{P} = \sum_{i=1}^{N} K_{Run,i}$ $= \mathcal{K} + K$ as well. This means that exactly two local units must be on at any sample until sample $\mathcal{K}$ and that exactly one local unit must be on at any sample after sample $\mathcal{K}$ in order for a solution with zero slack to exist. However, such a solution can exist if and only if there also exists a subset of $K_{Run,i}, i = 1, 2, \ldots, N$, which sums to $\mathcal{K}$.

This, however, corresponds exactly to the Subset-Sum Problem for the set $S = K_{Run,i}, i = 1, 2, \ldots, N$ and $t = \mathcal{K}$ since $\mathcal{K}$ and K are arbitrarily chosen positive integers. Thus, if there exists a polynomial time algorithm for solving the considered in-

stance of the DVPPDP then this algorithm could also solve the Subset-Sum problem in polynomial time. It now follows from Lemma 6 that the DVPPDP is NP-complete. □

CPLEX Performance

A commonly used option for solving integer problems is to use the software package CPLEX [18]. If CPLEX is capable of solving the Discrete Virtual Power Plant Dispatch Problem to optimality within a reasonable time frame, then there will be no reason to apply meta-heuristic methods to the problem. In this section we will therefore investigate how CPLEX handles the Discrete Virtual Power Plant Dispatch Problem.

We have tested CPLEX performance on ten *Portfolio(25,100)* and ten *Portfolio(50, 100)* problems. In many cases the computations are terminated with an error message stating that the computer has run out of memory and therefore no optimal solution is found. We observed that for *Portfolio(25,100)* five of ten problems were solved with an average computation time of 8 minutes and for *Portfolio(50,100)* zero of ten problems were successfully solved. Since all calculations finish due to lack of memory for 50 units and 100 samples, there is little hope that this option will scale to larger problem instances.

4 Meta-Heuristic Algorithms

Since we have illustrated that finding optimal solutions of the Discrete Virtual Power Plant Dispatch Problem is indeed very challenging, we will now investigate the performance of the meta-heuristic methods **Hill Climber**, [19], and **Greedy Randomized Adaptive Search Procedure**, [20].

Hill Climber

We first define a neighborhood associated with the Discrete Virtual Power Plant Dispatch Problem, which is based on the idea of an *n-move*. Next we develop two variations of the algorithm denoted *Uniform Selection Hill Climber* and *Weighted Selection Hill Climber*.

Neighborhood and n-move

Let an instance of the Discrete Virtual Power Plant Dispatch Problem be given, that is, a set of parameters $\overline{P}_i$, $K_{End,i}$ and $K_{Run,i}$, $i = 1, 2, \ldots, N$ and a dispatch sequence $P_{Dispatch}(k) \in \mathbb{R}, k = 1, 2, \ldots, K$. A solution of the Discrete Virtual Power Plant Dispatch Problem is then given by a set of feasible start times, $\boldsymbol{K_{Start}} = (K_{Start,1}, \ldots, K_{Start,N})$. The solution space is given by

$$\begin{aligned} S = \{&(K_{Start,1}, \ldots, K_{Start,N}) \in \mathbb{N}^N \\ &|K_{Start,i} < K_{End,i} - K_{Run,i}, i = 1, 2, \ldots, N\} \end{aligned}$$

and we have that

$$\begin{aligned} |S| &= \Pi_{i=1}^{N} K_{End,i} - K_{Run,i} \\ &\leq K^N. \end{aligned}$$

In the Discrete Virtual Power Plant Dispatch Problem a neighborhood map $\nu_n : S \rightarrow S^M$, defines for each solution $\boldsymbol{K_{Start}}$ a neighborhood set $S_n(\boldsymbol{K_{Start}}) \in S^M$ consisting of the set of solutions that can be obtained from $\boldsymbol{K_{Start}}$ by moving the start time of any n local units to feasible locations. This is called an n-move. In other words, a neighborhood map is a map of the form

$$\nu_n(\boldsymbol{K_{Start}}) = \{\boldsymbol{K_{Start}}' \in S \mid \boldsymbol{K_{Start}}' \text{ is obtained from } \boldsymbol{K_{Start}} \text{ by an } n\text{-move}\}.$$

Uniform Selection Hill Climber* and *Weighted Selection Hill Climber

Pseudo code for the **Hill Climber** method is given in **Algorithm 6**. The **Hill Climber** method first generates a random initial solution for the considered problem. Next a solution in the neighborhood of the current solution is found. If the cost of the neighboring solution is less than the cost of the current solution, then the neighbor solution will take its place as current solution. This procedure is continued until the time limit is reached.

Algorithm 6: Hill Climber$(\{LU_i\}_{i=1,2,\ldots,N}, \{P_{Dispatch}(k)\}_{k=1,2,\ldots,K}, n)$

1: Generate initial solution $\boldsymbol{K_{Start},_0}$ by randomly choosing feasible start samples for each local unit.
2: **Repeat**
3: Select $\boldsymbol{K_{Start}}'$ in $\nu_n(\boldsymbol{K_{Start},_i})$ by use of **Uniform Selection** or **Weighted Selection**
4: If $f(\boldsymbol{K_{Start}}') < f(\boldsymbol{K_{Start},_i})$ **then**
5: $\boldsymbol{K_{Start},_{i+1}} = \boldsymbol{K_{Start}}'$
6: **Until** time-limit is reached

Two tailored versions of the **Hill Climber** method, denoted *Uniform Selection Hill Climber* and *Weighted Selection Hill Climber*will be investigated.

In ***Uniform Selection Hill Climber*** the initial solution $\boldsymbol{K_{Start}}'$ is found by choosing n local units from the portfolio with uniform probability and then selecting feasible start samples for each local unit, also with uniform probability. Pseudo-code for **Uniform Selection** is given in **Algorithm 7**. Allowing n to be larger than 1 means that it is possible to accept a solution where a unit is moved to a less favorable start sample as long as other units are simultaneously moved to beneficial positions. This could help the algorithm escape a local optimum, which would not be possible if only one local unit can be moved at a time.

Algorithm 7: Uniform Selection$(\{LU_i\}_{i=1,2,\ldots,N}, n)$

1: **for** $j = 1$ **to** n **do**
2: Select LU_i from $\{LU_i\}_{i=1,2,\ldots,N-j+1}$ with probability $\frac{1}{N-j+1}$
3: Select start sample k for LU_i with probability $\frac{1}{K_{Start,i}}$
4: Set $\{LU_i\}_{i=1,2,\ldots,N-j} = \{LU_i\}_{i=1,2,\ldots,N-j+1} \backslash LU_k$
5: **end for**

In an alternative implementation, denoted ***Weighted Selection Hill Climber***, the n local units are again chosen uniformly from the portfolio, but the start time of each local

unit is now chosen with probability proportional to the benefit/cost of moving the start time of the local unit to each feasible time slot. Pseudo-code for **Weighted Selection** is given in **Algorithm 8**. If improving start times exist, we choose an improving time with probability proportional to the obtained benefit. On the other hand, if no improving start times exist, then we choose a deteriorating time with probability inversely proportional to the cost.

Algorithm 8: Weighted Selection$(\{LU_i\}_{i=1,2,\ldots,N}, \{P_{Dispatch}(k)\}_{k=1,2,\ldots,K}, n)$

for $j = 1$ **to** n **do**
  Select LU_i from $\{LU_i\}_{i=1,2,\ldots,N-j+1}$ with probability $\frac{1}{N-j+1}$
  for $k = 1$ **to** $K_{End,i} - K_{Run,i}$ **do**
    $v_k \leftarrow \Delta_{Cost}(i,k)$ given that $j-1$ local units have already been assigned new start samples.
  end for
  Construct $v_{negative}$ containing all negative numbers in v.
  if Length($v_{negative}$) ≥ 1 **then**
    Select start sample k for LU_i with probability $\frac{v_{negative}(k)}{\sum v_{negative}}$
  else
    Select start sample k for LU_i with probability $\frac{\frac{1}{v(k)}}{\sum \frac{1}{v}}$
  end if
  Set $\{LU_i\}_{i=1,2,\ldots,N-j} = \{LU_i\}_{i=1,2,\ldots,N-j+1} \backslash LU_k$
end for

Uniform Selection Hill Climber and ***Weighted Selection Hill Climber*** are tuned and compared in Section 5.

Greedy Randomized Adaptive Search Procedure

A definite weakness of the developed Hill Climber methods is that the initial solution is generated by choosing local units and start time with uniform probability. This means that the solutions generated for initial use have absolutely no similarity to $P_{Dispatch}$. This problem is mended in **Greedy Randomized Adaptive Search Procedure** (*GRASP*).

The idea in *GRASP* is to construct an initial solution one element at a time by use of a greedy algorithm. The choice of next element to be added is determined by constructing a candidate list of most beneficial choices. The probabilistic element in *GRASP* is then introduced by randomly choosing one of the candidates in the candidate list, but not necessarily the top candidate. After an initial solution has been generated, **Hill Climber** is called to achieve a further improvement of the solution.

Again two versions of the algorithm have been investigated, namely *GRASP Random* and *GRASP Sorted*. *GRASP Random* falls closest to the generic description of the *GRASP* algorithm, but as we will see later it has some challenges related to the Discrete Virtual Power Plant Dispatch Problem. To address this *GRASP Sorted* is developed as well.

Pseudo-code for *GRASP Random* is given in **Algorithm 9**. The idea is that m local units are chosen randomly from the portfolio and placed in the Unit List. Next the cost

Algorithm 9: ***GRASP Random*** $(\{LU_i\}_{i=1,2,\ldots,N}, \{P_{Dispatch}(k)\}_{k=1,2,\ldots,K}, m, l, n)$

repeat
  for $j = 1$ **to** N **do**
    for $k = 1$ **to** m **do**
      Unit List$(k) \leftarrow$ Select LU_i from $\{LU_i\}_{i=1,2,\ldots,N-j+1}$ with probability $\frac{1}{N-j+1}$.
      for $h = 1$ **to** $K_{Start,\text{Unit List}(k)} - K_{Run,\text{Unit List}(k)}$ **do**
        $v_{k,h} \leftarrow \Delta_{Cost}(\text{Unit List}(k), h)$ given that $j-1$ local units have already been assigned start samples.
      end for
    end for
    $CandidateList_{k,h} \leftarrow$ The l smallest entries in $v_{k,h}$.
    Select LU_k and start sample h from $CandidateList$ with probability $\frac{1}{l}$.
    Set start time of LU_k to h and set $\{LU_i\}_{i=1,2,\ldots,N-j} = \{LU_i\}_{i=1,2,\ldots,N-j+1} \backslash LU_k$.
  end for
  Hill-Climber$(\{LU_i\}_{i=1,2,\ldots,N}, \{P_{Dispatch}(k)\}_{k=1,2,\ldots,K}, n)$
until time-limit is reached

of starting each local unit in the Unit List at each feasible sample is computed and saved in v. The Candidate List is then generated by choosing the l smallest elements from v. Finally an element from the Candidate List is chosen with uniform probability.

When exploring *GRASP Random* it was discovered that for all problem sizes, *GRASP Random* generates initial solutions, which overshoot $P_{Dispatch}$ in the beginning of the horizon and fall lower than $P_{Dispatch}$ towards the end of the horizon. This behavior can be explained as follows:

Since slack is cheaper towards the end of the horizon *GRASP Random* will first start units early in the horizon as it builds an initial solution. At some point a decent match with $P_{Dispatch}$ is obtained for, say, the first 10 samples. However if a local unit then remains in $\{LU_i\}_{i=1,2,\ldots,N-j}$, which has deadline 10 or less, then it can only be added such that it makes the accumulated power consumption overshoot $P_{Dispatch}$ somewhere in the first 10 samples. When this has happened a number of times (see Figure 9.6) the result is a consumption profile, which overshoots $P_{Dispatch}$ in the beginning of the horizon and falls lower than $P_{Dispatch}$ towards the end of the horizon.

To alleviate this problem, the algorithm *GRASP Sorted* was developed. The algorithm is identical to *GRASP Random* except that local units are not initially chosen at random, but in sorted order, starting with the m local units with the earliest deadlines. The Candidate List is again generated based on the Unit List and an element from the Candidate List is chosen with uniform probability. Pseudo-code for *GRASP Sorted* is given in **Algorithm 10**.

5 Results

Before the algorithms can be compared they must be properly tuned. When applying a meta-heuristic there will always be a trade-off between time and performance, so compu-

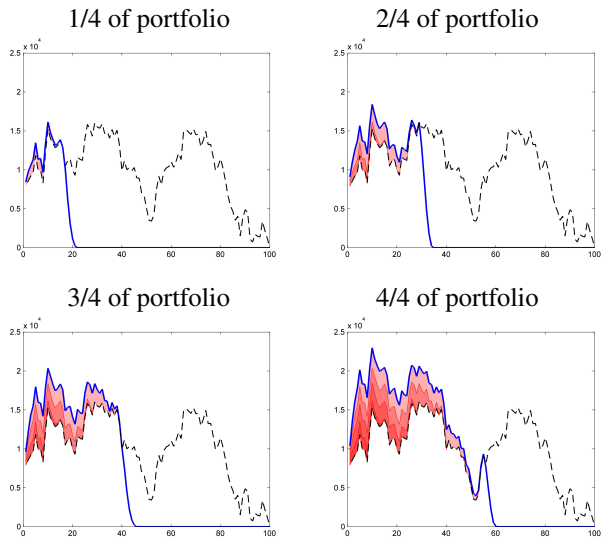

Figure 9.6: *GRASP Random* builds an initial solution one unit at a time and above it is depicted how the initial solution looks, when 1/4, 2/4, 3/4 and 4/4 of the units in the portfolio have been added.
Since slack is cheaper towards the end of the horizon, *GRASP Random* will first start units early in the horizon. When 1/4 of the portfolio has been added a decent fit with $P_{Dispatch}$ has been obtained for sample 0 to 18. However, there are still units remaining in the portfolio, which have deadline 18 or less, and now *GRASP Random* can only add these in such a way that the accumulated power consumption before sample 18 overshoots $P_{Dispatch}$ even further. This means that as units are added more and more positive slack builds up at the beginning of the simulation horizon, as can be seen from the progression of the figures.

tation time is fixed, not a parameter. All calculations are performed on a standard laptop. The algorithms have been implemented in C# [21].

Tuning

To tune the algorithms, a representative training test set of R problem instances is generated and each algorithm is run T times on each training data set. In order to be able to compare performance on problem instances with very different absolute values we compute the percentage gap and the percentage deviation. Since it is not feasible to determine the optimal solution of problems of the considered size, the percentage gap and the percentage deviation is computed relative to the minimum value found over all calculations on each particular problem instance.

In order to be able to compare performance on problem instances with very different absolute values, we compute the percentage gap $E = \frac{1}{S}\sum_{j=1}^{T}\frac{(z_j - z^*)\cdot 100}{z^*}$ and the

Algorithm 10: GRASP Sorted($\{LU_i\}_{i=1,2,\ldots,N}, \{P_{Dispatch}(k)\}_{k=1,2,\ldots,K}, m, l, n$)

repeat
  Sort $\{LU_i\}_{i=1,2,\ldots,N}$ according to $K_{End,i} - K_{Run,i}$.
  for $k = 1$ **to** m **do**
    Unit List(k) $\leftarrow LU_k$.
  end for
  for $j = 1$ **to** N **do**
    for $h = 1$ **to** $K_{Start,\text{Unit List}(k)} - K_{Run,\text{Unit List}(k)}$ **do**
      $v_{k,h} \leftarrow \Delta_{Cost}(\text{Unit List}(k), h)$ given that $j-1$ local units have already been assigned start samples.
    end for
    $CandidateList_{k,h} \leftarrow$ The l smallest entries in $v_{k,h}$.
    Select LU_k and start sample h from $CandidateList$ with probability $\frac{1}{l}$.
    Set start time of LU_k to h and set $UnitList = UnitList \backslash LU_k \cup LU_{j+1}$.
  end for
  Hill-Climber($\{LU_i\}_{i=1,2,\ldots,N}, \{P_{Dispatch}(k)\}_{k=1,2,\ldots,K}, n$)
until Time limit is reached.

Parameter	Test Values	Uniform Select. A	Uniform Select. B	Uniform Select. C	Weighted Select. A	Weighted Select. B	Weighted Select. C
$n-move$	{1, 5, 10, 50, 100, 500}	1	1	1	10	10	10

Table 9.2: The developed *Hill Climber* methods have one parameter, namely the value of n in $n-move$. To properly judge the performance of the algorithm a suitable value of this parameter must be found. This table states the best parameter value found for both *Hill Climber* methods.

percentage deviation $\sigma = \sqrt{\frac{1}{T-1}\sum_{j=1}^{T}\left(\frac{(z_j - z^*)\cdot 100}{z^*} - E\right)^2}$, where z^* is the optimal solution of the problem instance and j indexes the set of T candidate solutions. Since z^* is not known we will substitute the minimum value found over all calculations on the particular problem instance.

The tuning test is performed on randomly generated $Portfolio(N, 100),\ N = 10^3$, 10^4, 10^5, see Section 2. The tuning set consists of nine instances of the Discrete Virtual Power Plant Dispatch Problem (sets of portfolio and $P_{Dispatch}$) with three portfolios of 10^3, 10^4 and 10^5 local units each. Computation time is fixed at 10 seconds for all runs of the algorithms and $T = 10$. Results of parameter tuning test are given in Table 9.2 and 9.3 where **A** = 10^3 local units, **B** = 10^4 local units and **C** = 10^5 local units.

Parameter	Test Values	*GRASP Random*			*GRASP Sorted*		
		A	**B**	**C**	**A**	**B**	**C**
m	$\{1, 2, 3, 4\}$	1	1	1	4	4	4
l	$\{1, 2, 3\}$	3	3	3	1	1	2
Hill Climber-type	**Uniform** or **Weighted**	**W**	**W**	**U**	**W**	**U**	**W**
n	$\{1, 2, 3\}$	2	3	1	3	2	1
Time parameter for *Hill Climber*	$\{1, 2, 3, 4, 5, 8, 10, 15, 20\}$ times the time for generating initial solution	15	15	15	15	15	4

Table 9.3: The developed *GRASP* methods have the five parameters listed under 'Parameter'. This table states the best parameter value combination found for both *GRASP* methods.

Results

To finally test the algorithms a new $Portfolio(N, 100),\ N \in \{10^3, 10^4, 10^5\}$ is generated with three portfolios of 10^3, 10^4 and 10^5 local units each. Computation time is still fixed at 10 seconds for all runs of the algorithms and $T = 10$. The performance of all four algorithms is given in Table 9.4.

It is found that for all problem sizes the **Hill Climber** methods and *GRASP Random* have very similar performance, with no clear winner. *GRASP Sorted*, on the other hand, outperforms all the other methods by at least an order of magnitude both in terms of percentage gap and percentage deviation and for all problem sizes.

Figure 9.7, 9.8, 9.9 and 9.10 depict solutions found for a $Portfolio(100.000, 100)$ problem using each algorithm and the parameters given in Table 9.2 and 9.3. A visual inspection confirms the general conclusions that *Uniform Selection Hill Climber*, *Weighted Selection Hill Climber* and *GRASP Random* have very similar performance as the curves can hardly be distinguished. However, *GRASP Sorted* is clearly superior. It can be seen that particularly at the beginning of the simulations *GRASP Sorted* has less slack than the other methods and *GRASP Sorted* has furthermore found a far more sorted solution, which can be seen by the very steep slopes of the quarter lines in Figure 9.10.

As demonstrated in Section 3 there exists no efficient strategy for determining optimal solutions of the Virtual Power Plant Dispatch Problem for large problem instances. One option would be to generate artificial, structured problem instances where the optimal solution is known. However, inherent structure in a problem could likely favor one of the

	1.000 units		10.000 units		100.000 units	
	E	σ	E	σ	E	σ
Uniform Select.	32.4	7.7	58.6	3.6	47.8	1.5
Weighted Select.	38.0	8.6	62.5	4.5	59.3	1.7
GRASP Random	23.7	3.1	65.3	2.5	32.9	1.4
GRASP Sorted	0.6	0.4	2.5	0.3	0.7	0.5

Table 9.4: Percentage gap and percentage deviation for all developed methods.

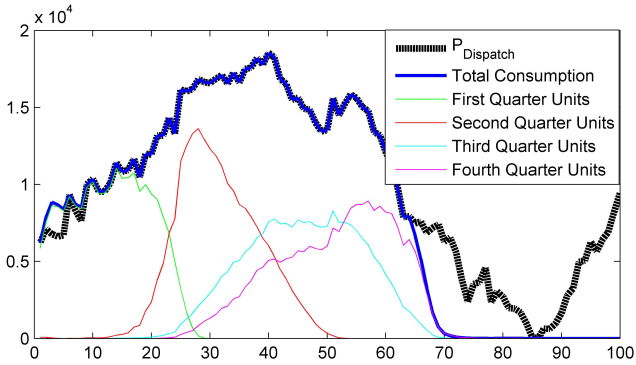

Figure 9.7: Performance of *Uniform Selection Hill Climber* for a $Portfolio(10^5, 100)$ problem.

methods, in particular *GRASP Sorted*, and would thus lead to unfair comparisons.

Our best available optimal solutions are therefore the five *Portfolio(25,100)* problem instances, which were successfully found by CPLEX in Section 3. In Table 9.5, *GRASP Sorted* has been retuned for *Portfolio(25,100)* problems and the average performance over ten solutions is given for these problems. It is found that the deviation from optimality is 6% to 13%, which is fairly good considering the short computation time.

6 Conclusion

In the vision for the future Smart Grid, not just hundreds, but thousands or even millions of flexible consumers must be coordinated to operate in a sensible, interconnected manner. A major issue in implementing the Smart Grid, however, is that this must happen in real time. In this paper we have therefore investigated computational speed of large scale dispatch problems.

Our investigations have concentrated on the Discrete Virtual Power Plant Dispatch Problem. Firstly the computational complexity was investigated by proving that the prob-

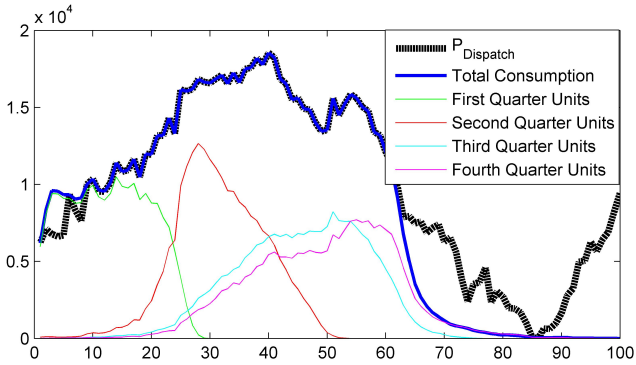

Figure 9.8: Performance of *Weighted Selection Hill Climber* for a $Portfolio(10^5, 100)$ problem

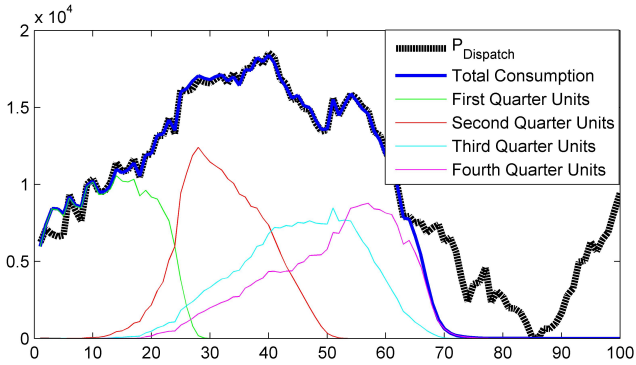

Figure 9.9: Performance of *GRASP Random* for a $Portfolio(10^5, 100)$ problem.

lem is NP-complete and investigating the options of solving the Discrete Virtual Power Plant Dispatch Problem by use of CPLEX [18]. Being NP-complete the Discrete Virtual Power Plant Dispatch Problem is therefore at least as complex (hard) to solve as the well-known Unit Commitment problem [23].

We therefore developed heuristic methods for solving the problem and specifically looked at **Hill Climber** and **Greedy Randomized Adaptive Search Procedure** (**GRASP**). Four algorithms were developed, denoted *Uniform Selection Hill Climber*, *Weighted Selection Hill Climber*, *GRASP Random* and *GRASP Sorted*. After tuning and testing, by far the best results where obtained by *GRASP Sorted*. This method can determine solutions, which are both agile (sorted) and well balanced even for problems of 100.000 units, 100 samples and with a computation time of just 10 seconds.

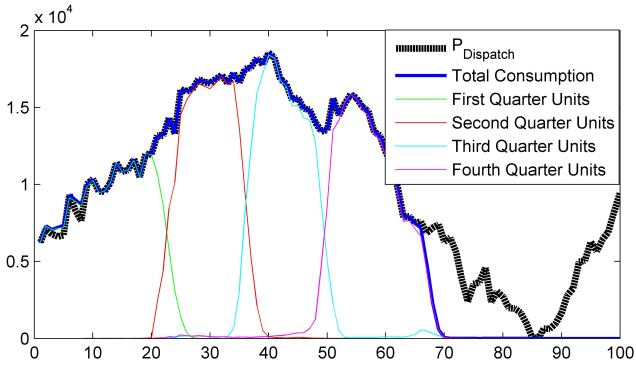

Figure 9.10: Performance of *GRASP Sorted* for a $Portfolio(10^5, 100)$ problem.

	Optimal	*GRASP Sorted*	Increase
Test 1	8224385	8755518	6%
Test 2	8509987	9208134	8%
Test 3	10881238	11893818	9%
Test 4	11326497	12614928	11%
Test 5	9763022	11055801	13%

Table 9.5: Optimal solutions values found in CPLEX and solution values found by *GRASP Sorted* for five *Portfolio(25,100)* problems instances.

References

[1] A. Mohsenian-Rad, V.W.S. Wong, J. Jatskevich and R. Schober and A. Leon-Garcia, *Autonomous Demand-side Management Based on Game-theoretic Energy Consumption Scheduling for the Future Smart Grid*, IEEE Transactions on Smart Grid - 2010, Volume 1, Issue 3, pp. 320–331

[2] Duncan S. Callaway and Ian A. Hiskens, *Achieving Controllability of Electric Loads*, Proceedings of the IEEE, Vol. 99, No. 1, January 2011, pp. 184–199.

[3] K. Edlund, J.D. Bendtsen and J.B. Jørgensen, *Hierarchical model-based predictive control of a power plant portfolio*, Control Engineering Practice, Vol. 19, pp. 1126–1136, 2011.

[4] K. Aoki, T. Satoh, M. Itoh, T. Ichimori, and K. Masegi, *Unit Commitment in a Large-Scale Power System including Fuel Constrained Thermal and Pumped-Storage Hydro*, IEEE Transactions on Power Systems, Vol 2(4), pp. 1077–1084, 1987.

[5] A. K. Ayuob and A. D. Patton, *Optimal thermal generating unit commitment*, IEEE Transactions on Power Apparatus and Systems, 90(4):1752–1756, 1971.

[6] Jing Huang, Vijay Gupta and Yih-Fang Huang, *Scheduling Algorithms for PHEV Charging in Shared Parking Lots*, 2012 American Control Conference, 2012, pp. 276-281.

[7] Shengbo Chen, Prasun Sinha and Ness B. Shroff, *Scheduling Heterogeneous Delay Tolerant Tasks in Smart Grid with Renewable Energy*, 51st IEEE Conference on Decision and Control, 2012, pp. 1130-1135.

[8] K. C. Sou, J. Weimer, H. Sandberg, and K. H. Johansson, *Scheduling Smart Home Appliances Using Mixed Integer Linear Programming*, 50th IEEE Conference on Decision and Control and European Control Conference, Orlando, pp. 5144–5149, 2011.

[9] D. T. Phan, J. Xiong and S. Ghosh, *A Distributed Scheme for Fair EV Charging Under Transmission Constraints*, American Control Conference, Montreal pp. 1053–1058, 2012.

[10] Z. A. Vale, H. Morais, H. Khodr, B. Canizes and J. Soares, *Technical and Economic Resources Management in Smart Grids using Heuristic Optimization Methods*, IEEE Power and Energy Society General Meeting, pp. 1–7, 2010.

[11] P. Faria, J. Soares, Z. Vale, H. Morais and T. Sousa, *Modified Particle Swarm Optimization Applied to Integrated Demand Response and DG Resources Scheduling*, IEEE Transactions on Smart Grid, Vol. 4, No. 1, pp. 606–616, March 2013.

[12] Z. Chen, L. Wu and Y. Fu, *Real-Time Price-Based Demand Response Management for Residential Appliances via Stochastic Optimization and Robust Optimization*, IEEE Transactions on Smart Grid, Vol. 3, No. 4, pp. 1822–1831, December 2012.

[13] T. Logenthiran, D. Srinivasan and T. Z. Shun, *Demand Side Management in Smart Grid Using Heuristic Optimization*, IEEE Transactions on Smart Grid, Vol. 3, No. 3, pp. 1244–1252, September 2012.

[14] S. Salinas, M. Li and P. Li, *Multi-Objective Optimal Energy Consumption Scheduling in Smart Grids*, IEEE Transactions on Smart Grid, Vol. 4, No. 1, pp. 341–348 March 2013

[15] P. Yi, X. Dong, A. Iwayemi, C. Zhou, and S. Li, *Real-Time Opportunistic Scheduling for Residential Demand Response*, IEEE Transactions on Smart Grid, Vol. 4, No. 1, pp. 227–234, March 2013.

[16] A. Subramanianz, M. Garcia, A. Domnguez-Garca, D. Callawayx, K. Poollay and P. Varaiyay, *Real-time Scheduling of Deferrable Electric Loads*, American Control Conference, pp. 3643-3650, 2012.

[17] *Introduction to Algorithms, Second Edition*, Thomas H. Cormen, Charles E. Leiserson, Ronald L. Rivest and Clifford Stein, The MIT Press, 2001.

[18] `www-01.ibm.com/software/integration/optimization/cplex-optimization-studio/`

[19] *Search Methodologies: Introductory Tutorials in Optimization and Decision Support Techniques*, Edmund K. Burke and Graham Kendall, Springer, 2005.

[20] Thomas A. Feo and Maurico G.C. Resende, *Greedy Randomized Adaptive Search Procedures*, Journal of Global Optimization 1995, No. 6, pp. 109–133.

[21] `www.msdn.microsoft.com`

[22] Saul I. Gass and Carl M. Harris, *Encyclopedia of Operations Research and Management Science*, Springer US, 1996.

[23] G. Xiaohong, Z. Qiaozhu and A. Papalexopoulo, *Optimization based methods for unit commitment: Lagrangian relaxation versus general mixed integer programming*, Proc. Of IEEE Power Engineering Society General Meeting, July 2003